

COVER PHOTO BY ROB BRIMSON

Electronics & Computing Monthly
Priory Court, 30-32 Farringdon Lane,
London, EC1R 3AU

Editorial 01-251-6222
Editor Gary Evans
Deputy Editor William Owen

Advertising 01-251-6222
Advertisement Manager Tony Herman
Advertising Executive Tracy Keighley
Advertising Production Serena Hadley

Production 01-251-6222
Art Editor Jeremy Webb
Make-up Time Graphics
Publisher Terry Pratt

Distribution
EMAP National Publications

Published by
EMAP Business and
Computer Publications

Printed by
Riverside Press, England

Subscriptions and Back Issues
Subscriptions
please telephone 01-251 6222
for details.
Back Issues telephone 0858 34567

Electronics & Computing Monthly is
normally published on the 13th day
of each month.

© copyright EMAP Business & Computer
Publications Limited 1985. Reasonable care is
taken to avoid errors in this magazine however,
no liability is accepted for any mistakes which
may occur. No material in this publication may
be reproduced in any way without the written
consent of the publishers. Subscription rates:
UK £15.00 incl. post. For overseas rates apply to
Subscription Dept., Priory Court, 30/32
Farringdon Lane, London EC1R 3AU.



ELECTRONICS & COMPUTING

Contents

VOL 5 Issue 6

SOFTWARE

Amstrad screen dump 20

A program for the computer of the moment. The software provides bit-image screen dumps from any of the CPC464's screen modes. Should work with the new 664 as well.

BBC disk file expander 65

Overcome the 31 file barrier presented by Acorn's DFS. The program works with both 40 and 80 track disks and caters for up to 58 files.

Building on QDOS 30

In the first part of a major new series, Adam Denning develops QSHELL, a collection of machine code routines that complement QDOS and provide the QL with the functions associated with conventional 16-bit operating systems.

Spectrum BASIC/machine code interface 45

A method of passing strings between BASIC programs and machine code routines, all with the greatest of ease.

Wordwise printer effects 66

The lack of standardisation of printer symbols and punctuation marks can lead to problems in producing the finished results. We show how these difficulties can quickly be overcome when using the Wordwise word processor.

PROJECTS

DIY plotter 24

Continuing our series describing the construction of a low-cost high performance plotter.

Video printer 42

The final part of our video digitiser project has details of the printer driver software.

Spectrum WP 50

This month we describe the hardware at the heart of the E&CM Spectrum wordprocessor. There's also some software that gets the system up and running.

FEATURES

COVER
FEATURE

QL - one year on 16

The QL became generally available one year ago. We take a look at how the hardware support industry for the computer has developed.

Making a mark 26

Mike James examines the concepts of mass storage systems with particular reference to the demands of home computer users. As the series develops the series will explain how to maintain storage systems and how to get the best performance out of the common forms of storage.

68 computing 35

All the latest news from the world of 68xx computing, including an exclusive interview with Eurohard's Development Manager and a look at OS9 command line separators.

Dragon 64 - the full circuit 37

Following the success of our analysis of the BBC micro's circuit diagram, we turn our attention to the Dragon 64 computer. The Dragon uses a number of interesting techniques including a very clever selection of silicon.

Comms corner 57

Latest comms news together with hints and tips for anyone contemplating writing their own terminal emulation software for the BBC micro.

REVIEWS

Quen Data DW1120 48

Low cost daisywheels now offer an alternative to dot matrix printers. This article assesses the performance of one of the latest of this breed of printer.

CPC664 55

The much leaked new machine from Amstrad is based on the familiar CPC464, the cassette deck has simply been replaced by a disk drive. Hardly the most innovative product, but one that is likely to make Amstrad even more profits.

Book reviews 61

The Hacker's handbook and Wiseowl's Hardware Guide to the BBC micro.

Acornsoft LOGO 70

Acornsoft's implementation of the LOGO language is very good value for money. Here we explain what you get for your cash.

PLUS

Editorial	10
News	10
Next Month	19
PCB service	33
Subscriptions	47
Book service	69

NEWS NEWS NEWS NEWS

Hackers under attack

In recent months the activities of hackers have received a good deal of coverage in the popular press. The Fleet Street hacks, in their approach to this subject, are in danger of promoting the computer hacker to the status of a modern day folk hero—the individual fighting against the Big Brother authority that large public/private data base systems represent. Popular TV series and some recent films have also been responsible for promoting this romantic view of the hacker. What is all too often overlooked is the fact that the actions of the hacker are at best a major irritation to those running data systems over the public telephone network and at worst are criminal.

As yet there is no clear-cut legal precedent covering hackers operating with simple home microcomputers in conjunction with a modem. All sorts of legal angles have been considered including such obscure ideas as applying the ancient law of trespass to computer systems. Recently though the authorities have turned to the 1981 Forgery and Counterfeiting Act in order to prosecute those indulging in the sport of hacking. A casual reading of the Act suggests that this particular piece of legislation includes the hacker among the ranks of those who try to pass dud cheques and funny money, and builds in some fairly stiff penalties that could put the hacker behind bars.

Although the application of this law to a hacker has yet to be put to the test in court, to the lay eye it provides a clear cut weapon to use against the hacker. The Act specifically includes information stored on disk or tape as being within the scope of material that may be fraudulently used and further states that it is possible to defraud a machine(!)

In this case it would appear that if, for example, a hacker caused a system to respond to a password which they were not authorised to use and that thus the computer was operating under the 'assumption' that the hacker was in fact the legitimate owner of the password, they would be in breach of the Act.

Even if the courts decided that the mere act of logging on to a system under an unauthorised password was not sufficient cause to convict, as soon as the hacker moves further into the data base things become much worse.

The Act defines a 'loss' that may result from a fraud as including 'not getting what one might get as well as parting with what one has'. As it is generally accepted that it is possible to steal information, any hacker viewing, say the content of someone else's mail box is depriving the owner of the mail box the exclusive access to the information held within it.

Things will become clear after a prosecution under this Act has been brought to court, but it would appear that the outcome won't be weighted in the hackers favour.

GARY EVANS

For additional comment on the subject of hacking see Communications News on page 57, and the review of the 'Hacker's Handbook' on page 61.

We would also be interested in hearing readers' views on the rights and wrongs of hacker's activities, write to The Editor at Priory Court, 30-32 Farringdon Lane, London EC1R 3HU.



Tandy PC almost breaks the one thousand pound barrier

The time when PC compatible computers will fall in price to a level at which they will be a serious option for a home user is not too far away. The recently launched Tandy 1000 just fails to break the £1000 barrier by a matter of £99. The basic machine comes complete with 128K of RAM and a single 5.25" disk drive. Software supplied with the computer includes MS-DOS version 2.11, BASIC and Deskmate an

integrated package which combines word processing, spreadsheet, file management, telecommunications, diary and electronic mail.

We'll have more on the Tandy 1000 and other 'business' machines that by virtue of their price can be considered for use in the home in an extensive feature in a couple of months time.

A cure for insecurity

The QL computer's microdrives offer an admirably low cost mass storage system, but have yet to convince people that they can have the reliability of conventional disk based systems. It would be a brave user who would commit valuable data to a micro drive without frequent use of the time consuming back-up process. The chance that files would be lost forever is all too great.

Anyone who has such worries will be interested in a new utility from Adder Publishing. QDOCTOR is a program that offers QL users the ability to recover corrupted data files. It performs this function by providing a menu driven range of utilities.

The program's VERIFY option

analyses the content of a microdrive cartridge in detail and can produce a print out that offers a complete analysis of any potential problem areas. QDOCTOR's full screen sector editor can then be used to recover any sectors that have become corrupted. Such editing can use either a 'HEX' or 'ASCII' window to allow both text and data files to be corrected with ease. In addition, options to recover from more serious problems are offered—it is even possible to recover files that have been deleted by mistake.

QDOCTOR is available from Adder Publishing, PO Box 148, CAMBRIDGE, CB1 2EQ. Price is £17.19 including VAT and postage.

NEWS NEWS NEWS NEWS

Commodore comes unstuck

When a company sees that sales of its product are not as brisk as it would like, or indeed when unit sales plummet to quite unacceptable levels the usual reaction is to slash the asking price.

Commodore cut the price of its new micros the Plus4 and the C16 a couple of months back and are now paying dearly for their actions. At least two major retail chains, Boots and Lasky, have reacted by saying they are unlikely to distribute any more Commodore products in the near future. This will mean that the Commodore 64 and the new C128 machine will not benefit from the High Street exposure that is crucial to the success of any computer. The stores have taken this action in order to avoid the possibility of being caught with large quantities of unsold stock which, in order to sell, must be offered at prices that are approaching cost.

Tatung has slashed the price of its Einstein computer by £150 to £350, and the computer is now selling as part of a £499 package. This consists of the Einstein itself, a monitor, BBC BASIC, word processor, spreadsheet and six games.

It is conventional wisdom that no company has yet successfully stimulated sales of a computer by slashing the price. The art is to enter the market at the 'right' price. Commodore's initial pricing of the Plus4 was the kiss of death, and it remains to see whether the Einstein will sell at its new low price.

Before leaving Tatung, there is news of a re-engineered version of the Einstein computer to be called the Micro Einstein. The major feature of this new design will be a custom ULA, to replace much of the discrete TTL found in the current model.

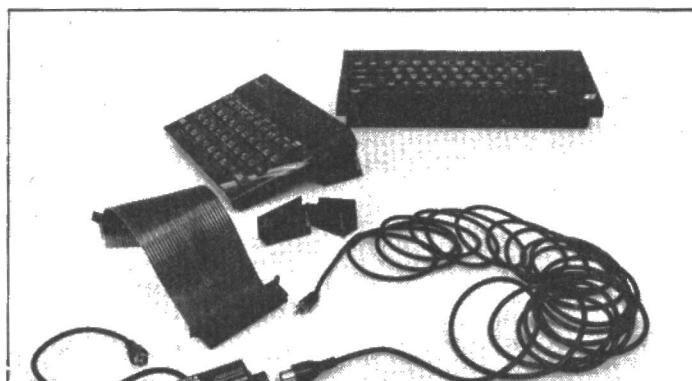
Not another QL ROM version!

Following hard on the heels of the revised versions of the Psion software bundled with the QL computer, the ROM based firmware of the machine has itself been updated. Many readers will recall that early versions of the computer were shipped with firmware that was so bug ridden as to be unworkable. In an effort to cure these faults, Sinclair has attempted various patches to the code and every week seemed to herald a new version of the QL ROM. It has been some time since the last - version JM - was released but this is now superceded

by a new ROM, version JS.

The new ROM fixes a number of the bugs present in JM but the major additions are to the way in which the QL handles errors. In addition some extra functions and other additions have been made to SuperBasic and most of the bugs evident in earlier versions of the interpreter have been tidied up. Sinclair now publish an official bug-list detailing many of the BASIC bugs and the ROMs in which they are present.

While an improvement over JM, ROM version JS is still far from the definitive version of the QL ROM.



Accessory to a Spectrum

Cheetah Marketing has launched a new range of low cost computer products. The additions to the company's established range of products include a Spectrum extender cable that extends the rear edge connector socket of both the Spectrum and Spectrum+. The idea is that this makes it easier to connect plug in modules to the machine. The price is £7.95.

A set of two computer feet costs £2.99 and can be used to angle the keyboard of any home micro to the optimum angle.

The other two products in the new range concern the RF video signal produced by home computers. The first of these is a £2.25 two-way aerial adaptor that avoids the continual lead swapping from TV broadcast reception to computer use. The adaptor makes it possible to have the computer and aerial leads connected to the TV set at the same time.

Cheetah's Aerial extension lead is simply a 15 foot extension lead that allows the computer to be used at a comfortable distance from the TV

Robotics convention

The first European Personal Robot Congress is to be held in London at the beginning of July. It is to be co-sponsored by the IEE, the British Personal Robot Manufacturers Group and the magazine Practical Robots, which was recently purchased by Oyez Scientific and Technical Services.

The three day event will comprise a conference which will look at the personal robot market, technical developments and at the industry in general. Speakers from Europe and the 'states' will be speaking in addition to some home grown robotists.

Running in parallel with the conference will be a series of specialist workshops and an exhibition designed to show the best in per-

sonal robotics. The UK finals of the Micromouse competition and the final heats of the World Robot Ping-Pong (should that be table tennis) competition will also be held at the same time.

To participate in all aspects of the Congress will cost delegates a hefty £250 + VAT but entry to the exhibition will be for a more affordable £1.50.

The Congress takes place between July 2nd to 4th at the West Centre Hotel in London. Any enquiries should be directed to:

Louise Marriot, Oyez Scientific and Technical Services Ltd, Bath House, 3rd Floor, 56 Holborn Viaduct, London EC1A 2EX. 01 236 4080.

CNC Lathe

The Colne 5 CNC Lathe is now available in a new Series II version. The Lathe is supplied either with software for the BBC micro (on disk and ROM) or the Commodore 64 (as a ROM cartridge). The lathe is capable of cutting mild steel and uses the standard ISO G-codes used by full-size machine tools.

Programming is menu-driven and an important feature of the software is that it enables toolpath emulation graphics creation. The lathe incorporates a variety of safety features - an important consideration for a device designed for use in educational establishments. These include a guard that covers all moving parts and a lockable 'panic' button that isolates the mains supply if activated.

screen. Coming soon, a pair of binoculars that allows you to read the screen.

Cheetah Marketing, Ray Street, LONDON, EC1R 3DJ. Telephone 01 933 4909.

A multitude of sins

SPEEDY EPROM BLOWER - JAN 85

The PCB overlay, Figure 4, should show a link in the lower right hand edge of the board. This completes the link between IC2 pin 9 and R7.

A number of readers have reported problems in the reliability of the operation of IC4, the 5 to 21V converter. These difficulties can usually be traced to the inductor L1. The 220uH value specified seems to work in some circuits though not in all. To verify that the 21V supply is reliable monitor the voltage at the cathode of D1 WHILE PROGRAMMING A BLANA EPROM. The voltage should remain constant at its nominal value. If this is the case the fault lies elsewhere in the circuit. Should the voltage fluctuate, you will need to change L1 for a different inductor, try a value of 470uH and preferably select an inductor of large physical size.

£40 MODEM - FEB 85

Some versions of the 2B modem require that pins 1 and 7 on the modem's 25 way D connector are linked together.

ON BREAK GOTO - MAY 85

The program line type set as

SET 1,(HL)

should in fact read

SET 7,(HL)

*Added to
ECS 1 MAY
1985 PG9*

AND FINALLY...

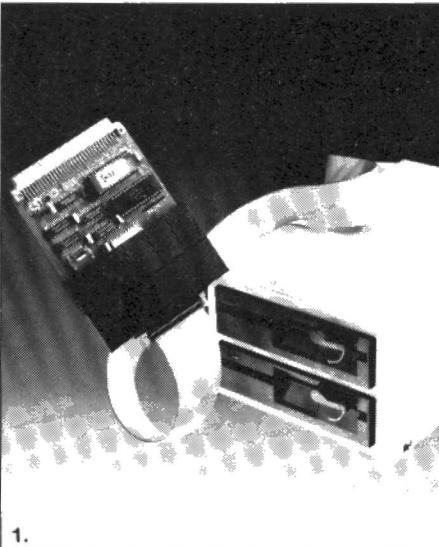
It has been brought to our attention that the price quoted for the Marconi Tracker Ball in last month's issue was incorrect. The correct price is £59.50 inclusive of VAT.

Our apologies to Central Trade Exchange, distributors of the Track Ball for any inconvenience caused by the error.

COVER STORY

Sinclair Research is celebrating one year of QL marketing (but only eight months of QL availability) with an extensive advertising campaign to sell the machine as an established contender with strong third party hardware and software support. E&CM has done its own investigation into the true strength of the QL peripheral market. David Green looks at the hardware.

Sinclair has failed to inspire a virile QL software market, mainly because of its rigid adherence to microdrives, coupled with an early policy of non-cooperation with software houses. The same mistakes haven't been made on the hardware side, not because Sir Clive's disciples have suddenly become more amenable, but rather as a result of pressure from several sources-bemused (and often 'not amused')



1.



2.

The QL industry who's who in hardware support

manufacturers, the media, and users – which has led to the fixing of standards for a wide range of interfaces and peripherals. The mysteries and idiosyncrasies of QDOS and QL I/O have been revealed to companies trying to produce QL add-ons which will work within the low tolerance levels required.

Sinclair itself is yet to produce a single hardware addition to the QL, but many third party suppliers have succeeded in doing so. The QL is a versatile animal and would-be peripherals have a choice of ports into which they can be connected: serial ports 1 and 2, control ports 1 and 2.

The RGB socket and the 64-pin (2 x 32) edge connector (the networking and ROM ports have yet to be utilised, though we suspect that will soon change).

With the exception of the control ports, which facilitate joystick input (specially wired leads for Atari-type joysticks are available), each of the three remaining I/O ports are pretty fussy about what gets plugged in. The number of companies pro-

ducing hardware for the QL at the moment is small because most equipment has to be QL-specific and won't work with other micros; this being the case it is vital that QL owners know about as many of the products that are currently available and which are the best buys.

Red green and blue

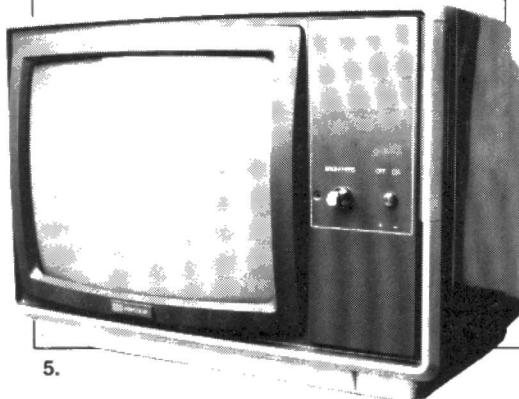
The only satisfactory method of utilising the full 80-column (or 85 column to be more accurate) display on the QL is to purchase an RGB monitor which, not surprisingly, plugs into the RGB socket on the back of the machine. There are at present four companies making monitors for the QL: Microvitec, Opus, Centel and Data Efficiency (Taxan-Kaga).

There is not much to choose between these four, with the proviso that the machine supplied by Opus has the appearance of an old TV set. The Microvitec is a modified version of their popular BBC monitor and is fairly close in looks to Cen-

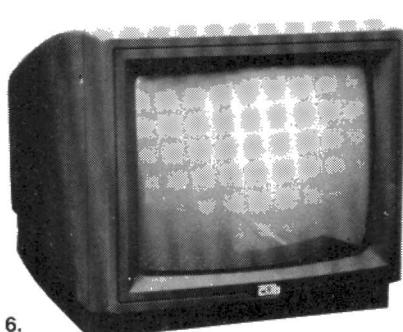
How many ways can you build up a QL system? David Green takes a hard port-by-port look at QL peripherals

tel's offering – chunky, squarish and matt black. The dark horse of this quartet, however, could be Data Efficiency's 'Vision QL' which sports the Sinclair badge as their official monitor (though whether the royalty charged for this seal-of-approval is recouped remains to be seen).

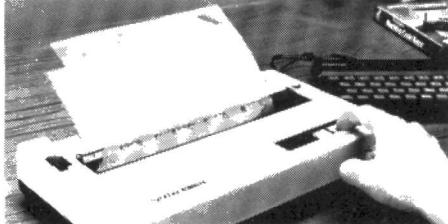
One last point before moving away from monitors: it's a good idea to check that the monitor you intend to purchase comes complete with connecting lead – wiring one up is a bit of a nightmare!



5.

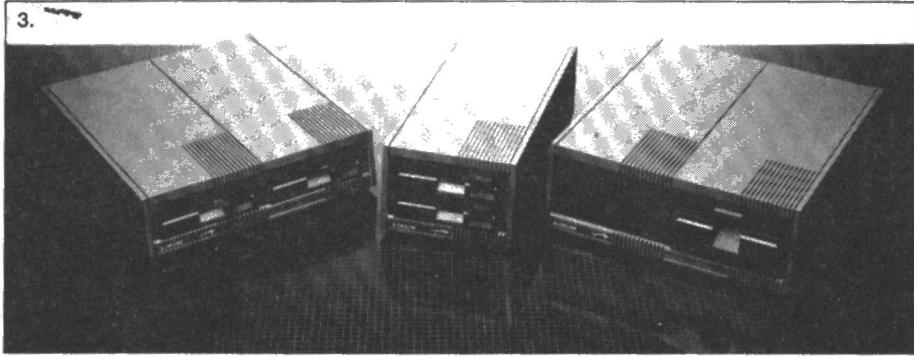


6.

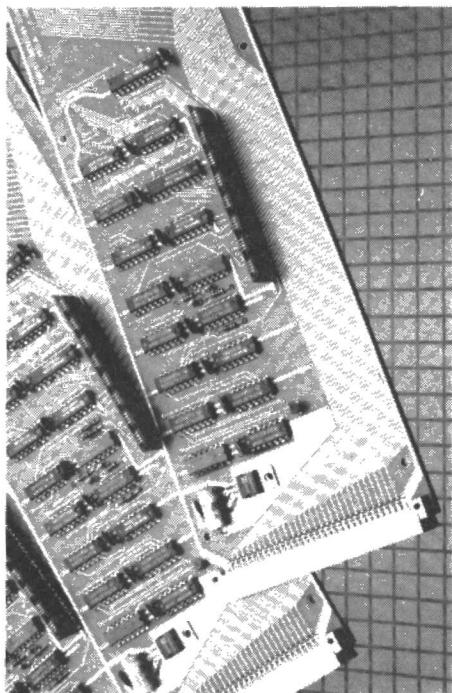


7.

5. The Opus RGB Monitor. 6. Microvitec's Cub QL Monitor. 7. The Brother HRS QL compatible printer.



1. The Computamate disk interface and 5.25" drives.
 2. MicroPeripherals' disk interface and 3.5" drives.
 3. The Quest QL 'Executive' series, includes both floppy and hard disk units.
 4. A mock-up of Sinclair's 0.5 Megabyte RAM extension was shown at the launch. Where is it today?



The Quest Executive Expansion card gives up to 512K extra RAM.

Serial or parallel?

Despite vociferous press criticism of various computer manufacturers' interpretations of the RS232 standard (!), machines are still being produced with their own quaint variations – the QL, unfortunately, is no exception. The two serial ports on the back are not only wired in a unique and confusing way (SER1 and SER2 transmit/receive lines are transposed, though Baud rates cannot be set at two different values), they are also difficult to control precisely without recourse to assembly level QDOS.

The easy solution is to forget about serial printers (and any other serial devices) and plump for one of the Centronics interfaces on the market along with a parallel printer. The choice of printer is not then restricted to QL compatibles and only leaves the interface to consider.

As with the monitors, there are several suitable devices on the market offering surprisingly few variations on the basic theme of small black boxes with permanently-fixed leads. The more sophisticated units offer a choice of Baud rates (eg, Care Electronics' serial-to-parallel convertor offers switchable rates from 75 to 9600 Baud), but since the QL defaults to 9600 Baud it seems sensible (if your printer can cope) to go for the cheapest fixed rate interface as provided by companies like MicroPeripherals or Miracle Systems.

For those of a more adventurous nature, the thrill of interacting with the QL's serial I/O directly may be too tempting to resist. This course, however, is laced with traps

Manufacturers and suppliers

MONITORS

Citadel Products	01 951 1848
MBS Data Efficiency	0442 60155
Microvitec	0274 390011
Opus Supplies Ltd	01 701 8668
Strong Computer Systems	0267 231246

Technomatic Ltd	01 208 1177
Zeal Marketing Ltd	0246 208555

PRINTERS

Datasystems	01 482 1711
MicroPeripherals	0256 473232
Microworld	0293 545630
Printerland	0484 514105
Strong Computer Systems	0267 231246

Technomatic Ltd	01 208 1177
Twickenham Computer Centre	01 891 4991

Viglen Computer Supplies	01 843 9903
Zeal Marketing Ltd	0246 208555

INTERFACES

Cambridge Systems Technology	0223 323302
Care Electronics	0923 777155
Computamate Data Products	0782 811711

Miracle Systems Ltd	0272 603871
Sigma Research	231 Coldhams Lane, Cambridge

Technology Research Ltd	0784 63547
-------------------------	------------

DISK SYSTEMS

Computamate Data Products	0782 811711
CST	0223 323302
Medic Datasystems Ltd	0256 52703
MicroPeripherals	0256 473232

Quest	04215 66488
-------	-------------

MEMORY EXPANSIONS

Eeprom Services	0532 667183
PCML Ltd	0372 67282/68631
Quest	04215 66488
Simplex Data Ltd	01 575 7531

MISCELLANEOUS

Action Computer Supplies	01 903 3921
Mains spike eliminator	

Eidersoft	01 478 1291
Quicksort II Joystick	

Management Science Ltd	17 West Hill, London SW18
QL case	

Power International	0705 756715
Mains spike eliminator	

Sigma Research	231 Coldhams Lane, Cambridge
Joystick adaptor	

Sinclair Research	0276 686100
Power supplies	

Transform Ltd (Dept QL)	089 283 4783
QL dust cover, microdrive storage box, RS232 lead	
Viglen Computer Supplies	

Printer stand	01 843 9903
Zeal Marketing Ltd	0246 208555

Printer peripherals

COVER STORY

for the unwary and when buying a serial printer it pays to follow the guiding principle: if it's not a dedicated QL printer and you can't find any information about its QL compatibility – don't buy it. Of all the serial printers on the market, the only ones I've had consistent success in connecting directly to the QL are the Epson, Brother, Quen-Data and Star Delta; though it must be said that even with these there's a lot of fiddling with DIP switches before everything works as expected.

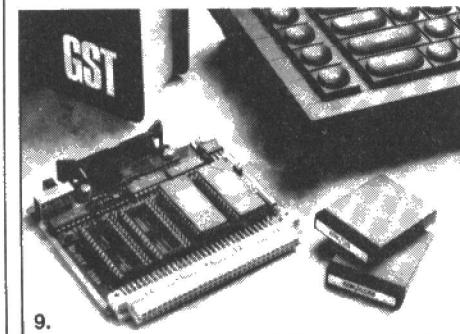
Ending it all

The really big news as far as QL owners are concerned is the arrival of the long-awaited disk drives and associated interfaces which plug neatly into the slot at the end of the machine.

First to deliver the goods were Computamate, who market the Q-Disc interface made by CST. This comes with optional dual Teac drives (double or single sided), but will also function with any Shugart compatible units. The interface consists of a small PCB (105 x 145mm) with an edge connector at one end and a matt black housing at the other to match the styling of the QL. Connection to the drives is via a one metre length of ribbon cable.

Using the system it soon becomes apparent that your microdrives will be demoted to providing secondary back-up. Access times on disk are at least ten times faster and this transforms Psion's software into the fully-fledged business packages they were originally intended to be.

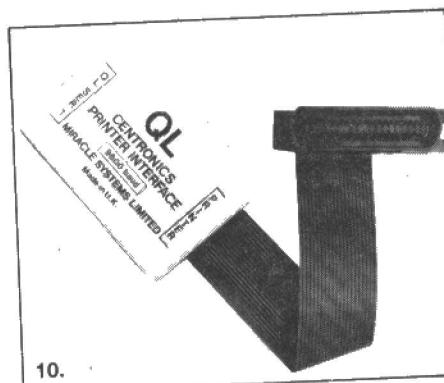
The Q-Disc system includes a barrage of extra SuperBasic commands, which bear a remarkable resemblance to those found in the QL Toolkit. This is no real surprise, since Toolkit's author (Tony Tebby) also had a hand in the development of the disk system. These additional commands provide such enhancements as group file operations, paged directories, file specification, number base conversion and default drive settings.



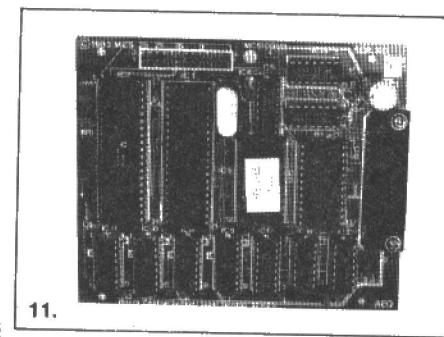
9. The GST 68K/OS board.

The only aspect of Computamate's system which lets it down is the manual, which is a very brief affair and does not describe any of the aforementioned commands. However, we have been assured that this is only a temporary arrangement and that a new manual is on the way.

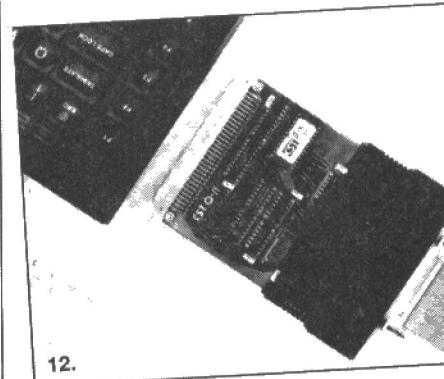
Next up on the disk drive bandwagon are MicroPeripherals with an interface of simi-



10.



11.



12.

10. Miracle Systems Centronics printer interface. 11. Microcontrol systems 32K buffer/printer interface. 12. CST parallel interface.

lar appearance and dimensions to the CST board. Where MP's version differs is in their selection of two neat 3.5" drives, making the complete system very compact indeed.

Along with selected 'Toolkit-like' commands – the result of collaboration with Sinclair rather than any contact with Mr Tebby – MicroPeripherals has written a number of special utilities including a microdrive emulator which allows any software that normally runs on microdrive to run on disk without recourse to reconfiguration software. Additional error messages are another feature of MP's EPROM-based extensions and we are told that updated ROMs will include many others.

The third company to have a disk interface and drives on the market is Quest. Taking a leaf out of the 'Sinclair Book of marketing Strategy', no doubt, advance publicity for their units preceded the launch by about six months, but now the drives have finally arrived some critics are asking what took so long. Both the interface and drives are about one third as big again as the nearest rival units (Computa-

mate), which makes carrying them around an alternative to aerobics!

The reason for all this extra bulk is partly due to Quest's decision to make all their equipment compatible with CP/M 68K – a bold move and one which could prove their downfall if the promised proliferation of 68K software is not forthcoming – and partly due to the inclusion of a cooling fan with the drives (perhaps a better PSU would have been a wiser move?).

Another drawback with the Quest system is that the disk boot-up software is on microdrive rather than EPROM and though it only takes 20 seconds or so to run it's still annoying having to wait each time you power up the drives. On the plus side, however, Quest have gone to the effort (and expense) of acquiring the Sinclair badge and making their system 'officially approved'.

There is one other disk system which warrants mention, more because of the publicity it has received rather than any specific capabilities. Medic Datasystems purport to produce one of the cheapest and most comprehensive disk systems for the QL (though so far nobody has been able to confirm their claims). If everything checks out, QL owners can look forward to a system with built-in RAM and instant switching between the Psion packages.

The item conspicuous by its absence from this list is OEL's QCOM modem. Much to the embarrassment of Sinclair Research, who advertised the modem heavily, OEL has gone into receivership. Sinclair says it will ensure the product can be bought by QL owners.

Extended or unextended

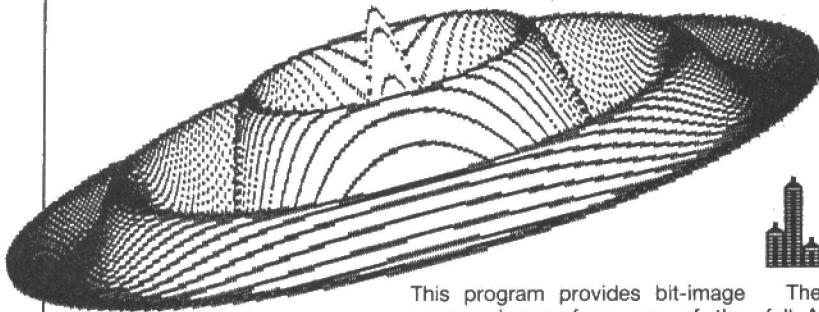
At least one disk system manufacturer has noted the importance of extra memory – in the form of plug-in RAM – concurrent with their DOS. This company, however, would appear to be the exception and most have opted for discrete RAM extensions. Two examples are Simplex Data and Quest, which both produce RAM extension boards for the QL. Quest's apparent obsession for 'big equals best' means that their extension is approximately twice the size of Simplex's; this makes it an unwieldy affair and troublesome to accommodate securely.

The Simplex board is a much neater arrangement, and its edge fits flush with the end of the QL. It is available in 256K and 512K byte sizes and with optional RAM disk software to allow instant access to stored programs (eg, Psion's packages).

Another company producing extra memory for the QL (under the rather confusing nom de plume of QL+) is PCML. Their boards come in sizes ranging from 64K to 256K and also include the aforementioned optional RAM disk utilities.

All these extensions function in a similar way when attached to the QL, with the only visible effect of connection being an increased time delay from power up to the F1/F2 selection screen.

AMSTRAD Screen dump



A bit-image screen dump for the Amstrad CPC464 by Gary Nugent.

This program provides bit-image screen dumps from any of the screen modes on the Amstrad CPC464. The program can produce two sizes of dump, however, the dumps are not shaded copies related to the ink colours on the screen: each pixel has the same intensity in the screen dump, regardless of its screen colour.

The smaller of the two dumps is approximately 135mm by 70mm in size, and the screen is copied from top to bottom. Three of these dumps can comfortably fit on an A4 page.

The larger of the dumps requires a full A4 page. Here the screen is copied from the left hand side to the right hand side, so the screen is effectively rotated through 90 degrees.

Both dumps are centered horizontally on the page, but not vertically. The dumps will not accurately reflect what is in the horizontal and vertical spacings of the pins in the print head. As a result the smaller dump is compressed and the larger dump is slightly elongated. The actual horizontal to vertical ratio of the screen is 1.6:1. The smaller

dump has a ratio of 1.92:1, and the larger dump a ratio of 1.33:1.

The program also contains an invert option. This effectively means that unprinted pixels will be printed, and plotted pixels ignored. In general this will produce a white on black dump. However, this type of dump wears out a ribbon rather quickly, and may be better employed to "re-invert" an inverted, or heavily plotted screen.

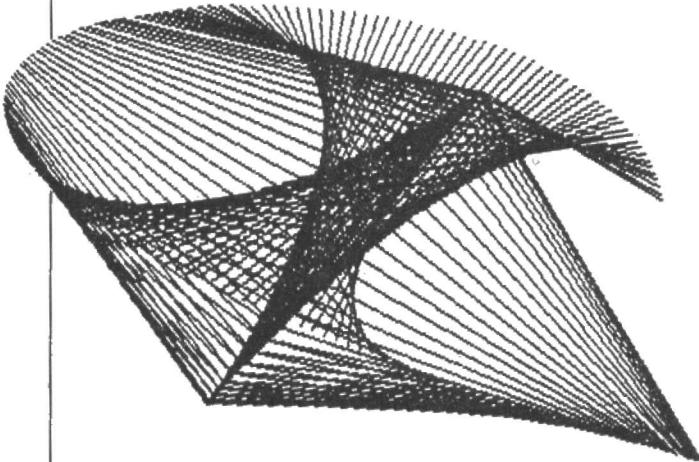
I decided to make this program into a Resident System Extension (RSX), (details can be found in the Amstrad Firmware Specification).

LISTING 1.

```

A2001 PRINT: CALL BD2E      A2681 PUSH HL      E5      A2D3: LD BC, #0007    01 07 00
A2031 JR C,PRINT      A2691 LD A, #42      3E 42  A2D6: AND A      A7
A2051 CALL BD2B      A26A1 CALL BB1E      CD 1E BB  A2D7: EX DE, HL    EB
A2081 RET      A26B1 POP HL      E1      A2D8: SBC HL, BC  ED 42
A2091 BITGRA      FD E5      A271: JR NZ,END1  20 11  A2DA: EX DE, HL    EB
A20B1 LOOP1: LD A, (IY+00)  A272: LD A, #0A      3E 0A  A2DB: LD A, E      7B
A20E1 CALL PRINT      A273: CALL PRINT      CD 00 A2  A2DC: AND D      A2
A2111 INC IY      FD 23      A276: LD BC, #000E  01 0E 00  A2DD: CP #FB      FE FB
A2131 DEC C      0D      A277: AND A      A7      A2DF: JR NZ,LOOP8  20 AB
A2141 JR NZ,LOOP1  20 F5      A278: SBC HL, BC  ED 42  A2E1: END2: RET      C9
A2151 POP IY      FD E1      A27C1: LD A, L      7D      A2E2: INIT: LD IY, #AA04  FD 21 04 AA
A2161 LD A, #00      3E 00      A27D: AND H      A4      A2E3: PUSH AF      F5
A2181 CALL PRINT      A27E: CP #F9      FE F9  A2E7: LD DE, #0000  11 00 00
A21A1 LOOP2: CALL PRINT      A280: JR NZ,LOOP5  20 A1  A2EA: LD HL, #0000  21 00 00
A21D1 DJNZ LOOP2  10 FB      A281: PUSH DE      DS      A2ED: CALL BBC9  CD C9 BB
A21F1 RET      C9      A282: END1: RET      C9      A2F0: LD A, #1B      3E 1B
A2201 SMALL: LD HL, #01BF  21 0F 01  A283: LARGE: LD DE, #027F  11 7F 02  A2F2: LD (IY+00), A  FD 77 00
A2231 LOOP5: LD BC, #A004  01 04 A0  A284: LOOP8: LD BC, #5004  01 04 50  A2F5: CALL PRINT  CD 00 A2
A2261 CALL BITGRA  CD 09 A2  A285: CALL BB1E  CD 09 A2  A2F8: LD A, #31      3E 31
A2291 LD DE, #0280  11 B0 02  A286: LD HL, #01BF  21 0F 01  A2F9: CALL PRINT  CD 00 A2
A22C1 LOOP4: PUSH HL      E5      A287: PUSH DE      DS      A2F4: LD A, #4C      3E 4C
A22D1 LD B, #07      06 07  A290: LD B, #07      06 07  A2F5: LD A, #03      3E 03
A22F1 LD C, #00      0E 00  A291: LD C, #00      0E 00  A2F6: LD (IY+01), A  FD 77 01
A2311 LOOP3: PUSH HL      E5      A294: LOOP6: PUSH DE      DS      A302: LD A, #03      3E 03
A2321 PUSH DE      D5      A295: PUSH HL      E5      A304: LD (IY+03), A  FD 77 03
A2331 PUSH BC      C5      A296: PUSH BC      C5      A307: POP AF      F1
A2341 PUSH HL      E5      A297: CALL BBFO  CD F0 BB  A308: CP #03      FE 03
A2351 LD HL, #0280  21 B0 02  A298: POP BC      C1      A309: RET NC      D0
A2381 AND A      A7      A299: AND A      A7      A30B: CP #00      FE 00
A2391 SBC HL, DE  ED 52  A29E1: JR Z,LDCARRY2  2B 01  A30D: JR Z, SETPT  2B 2B
A23B1 EX DE, HL    EB      A2A01: SCF      37      A30F: LD C, (IY+00)  DD 4E 00
A23C1 POP HL      E1      A2A1: LDCARRY2  RL C      A312: LD D, (IY+01)  DD 46 01
A23D1 CALL BBFO  CD F0 BB  A2A3: POP HL      E1      A315: CP #01      FE 01
A2401 POP BC      C1      A2A4: POP DE      D1      A317: JR Z,CHKINV  2B 1B
A2411 AND A      A7      A2A5: DEC DE      1B      A319: LD A, (IY+02)  DD 7E 02
A2421 CP #00      FE 00  A2A6: DZNZ LOOP6  10 EC  A31C1: OR (IY+03)  DD 86 03
A2441 JR Z,LDCARRY1  2B 01  A2A8: LD A, C      79      A31F: JR Z,NOTINV  2B 02
A2461 SCF      37      A2A9: XOR (IY+04)  FD AE 04  A323: NOTINV: LD IY+04, A  FD 77 04
A2471 LDCARRY1: RL C      CB 11  A2AC: LD C, A      4F      A326: LD A, #70      3E 70
A2491 POP DE      D1      A2AD: LD A, E      7B      A328: LD (IY+02), A  FD 77 02
A24A1 POP HL      E1      A2AE: AND D      A2      A32B: LD A, C      79
A24B1 DEC HL      2B      A2AF: CP #FB      FE FB  A32C: OR B      B0
A24C1 DEC HL      2B      A2B1: JR NZ,NOTE052  20 04  A32D: JR Z, SETSMALL  2B 0E
A24D1 DJNZ LOOP3  10 E2  A2B3: LD A, #70      3E 70  A32F: CALL NZ,LARGE  D4 B3 A2
A24F1 LD A, C      79  A2B5: AND C      A1      A332: JR RESET      1B 11
A2501 XOR (IY+04)  FD AE 04  A2B6: LD C, A      4F  A334: CHKINV: LD A, C      79
A2551 LD C, A      4F  A2B7: NOTE052: LD A, C      79  A335: OR B      B0
A2541 LD A, L      7D  A2B8: CALL PRINT      CD 00 A2  A336: JR Z, SETPT  2B 02
A2551 AND H      A4  A2B9: LD A, C      79  A339: LD A, #7F      3E 7F
A2561 CP #F9      FE F9  A2Bc: CALL PRINT      CD 00 A2  A33A: SETPT: LD (IY+04), A  FD 77 04
A2581 JR NZ,NOTE051  20 04  A2Bf: DEC HL      2B  A33D: SETSMALL: LD A, #20      3E 20
A25A1 LD A, #7B      3E 7B  A2C0: LD A, L      7D  A33F: LD (IY+02), A  FD 77 02
A25C1 AND C      A1  A2C1: AND H      A4  A342: CALL SMALL  CD 20 A2
A25D1 LD C, A      4F  A2C2: CP #FF      FE FF  A345: RESET: LD A, #0D      3E 0D
A25E1 NOTE051: LD A, C      79  A2C4: POP DE      D1  A347: CALL PRINT  CD 00 A2
A25F1 CALL PRINT      CD 00 A2  A2C5: JR NZ,LOOP7  20 CB  A34A: LD A, #1B      3E 1B
A2621 DEC DE      1B  A2C7: LD A, #42      3E 42  A34C: CALL PRINT  CD 00 A2
A2631 LD A, E      7B  A2C9: CALL BB1E  CD 1E BB  A34F: LD A, #32      3E 32
A2641 OR D      B2  A2C: JR NZ,END2  20 13  A351: CALL PRINT  CD 00 A2
A2651 POP HL      E1  A2CE: LD A, #0A      3E 0A  A354: RET      C9
A2661 JR NZ,LOOP4  20 C4  A2D0: CALL PRINT      CD 00 A2

```



These commands are called by preceding the chosen command name with a vertical bar (|). The command name for this program is :SCRDUMP. In addition, parameters may be passed to it. The command has the form:

:SCRDUMP,i

where "i" specifies the invert option: 0=normal, 1=invert
"s" specifies the dump size:
0=small, 1=large

I have left space for other extension commands to be added at some later date.

I own a Star Gemini 10X printer, for which this program was written. All the control codes used are Epson compatible. However, the line feed code sent at the end of each line may cause problems on the Epson. Character 13 (instead of character 10) might suit these printers better. The locations into which this value should be poked are &A272 and &A2CF.

Now to the program entry. Listing 5 should be entered first, and saved to tape with:

SAVE "CMD-LOADER"

Next enter Listing 4. This is a hex-loader which displays the hex-numbers as they are entered. You can also check the numbers that you have entered. The machine code in Listing 1 is entered now. It is advisable to save the code entered at this stage with:

SAVE "MC",B,&A200,350

Now enter the machine code in Listing 2 and Listing 3. Note the start addresses of these, (&AA00 and &AA46). Save all the machine code now, after "CMD-LOADER" with:

SAVE "COMMANDS",B,&A200,
2400

When "CMD-LOADER" is loaded and run, the user will be asked to press PLAY to load the next program ("COMMANDS"). The screen dump program will be initialised and a menu of the extension commands loaded will be printed on the screen. To leave the program, press Escape twice. The dump routines may now be called as indicated above.

LISTING 2.

AA00:	DEFB 4	00 00 00 00
AA04:	DEFB 4	00 00 00 00
AA08:	DEFB 2	00 00
AA0A: INITRSX	LD BC, #AA14	01 14 AA
AA0D:	LD HL, #AA00	21 00 AA
AA10:	CALL BCD1	CD D1 BC
AA13:	RET	C9
AA14: SPECADR	DEFB 2	46 AA
AA16: JFBLK	JP A2E2	C3 E2 A2

LISTING 3.

AA46: NAMETBL	DEFB 4	53 43 42 44
AA4A:	DEFB 4	55 4D 00 00

LISTING 4. Hex loader.

```

10 REM *****
20 REM |
30 REM | Amstrad Hex Loader |
40 REM |
50 REM | (C) Copyright Gary Nugent |
60 REM |
70 REM *****
80 MEMORY $41FF:MODE 1:BORDER 3:INK 0,0:INK 1,6:INK 2,24:INK 3,11
90 WINDOW #1,2,39,2,6:WINDOW #2,14,28,10,24:WINDOW #3,3,37,12,12
100 MOVE 15,385:DRAW 425,385,2:DRAW 625,303:DRAW 15,303:DRAW 15,385
110 MOVER -4,4:DRAW 628,389:DRAW 628,299:DRAW 11,299:DRAW 11,389
120 MOVE 206,257:DRAW 449,257:DRAW 449,14:DRAW 206,14:DRAW 206,257
130 MOVER -4,4:DRAW 452,261:DRAW 452,10:DRAW 202,10:DRAW 202,261
140 PAPER 0:OPEN 2:PAPER #1,1:PAPER #2,3:OPEN #3,1:CLS #2
150 LOCATE #1,8,3:OPEN #1,0:PAPER #1,2:PRINT #1," Start Address ":"PEN #1,2:PAPER #1,1
160 INPUT #1," ";a$:adr=VAL("6"+a$):adr1=adr
170 CLS #1:PEN #2,0
180 PRINT #2," ";HEX$(adr);":"
190 INPUT #2,x$:IF UPPERS(x$)="S" THEN 290
200 IF LEN(x$)=0 OR (LEN(x$)=1 AND UPPERS(x$)>"S") THEN PRINT #2,CHR$(11):GOTO 180
210 x=VAL("6"+x$):POKE adr,x
220 adr=adr+1
230 IF (adr-adr1)<0 THEN 280
240 PRINT #1,HEX$(adr-B1);":":FOR i=adr-B TO adr-1:x$=HEX$(PEEK(i))
250 IF LEN(x$)=1 THEN x$="0"+x$ 
260 PRINT #1," "+x$;NEXT i:PRINT #1
270 adr1=adr
280 GOTO 180
290 CLS:PRINT #3," Do you want to check your input ? "
300 b$=UPPERS(INKEY$):IF b$>"Y" AND b$<"N" THEN 300
310 IF b$="N" THEN CLS:GOTO 410
320 CLS:OPEN 0:WINDOW #2,2,39,2,24:PAPER #2,1:PEN #2,2:CLS#2
330 MOVE 0,0:DRAW 0,395,3:DRAW 639,399:DRAW 639,0:DRAW 0,0
340 MOVER 4,4:DRAW 4,395:DRAW 635,395:DRAW 635,4:DRAW 4,4
350 a1=VAL("6"+a$):a2=adr-1:FOR i=a1 TO a2 STEP 8:PRINT #2,HEX$(i+1);":":FOR j=0 TO 7
360 x$=HEX$(PEEK(i+j)):IF i+j>adr-1 THEN x$="00"
370 IF LEN(x$)=1 THEN x$="0"+x$ 
380 PRINT #2," "+x$;
390 NEXT i:PRINT #1
400 FOR i=1 TO 500:NEXT i:LOCATE 1,24
410 END

```

LISTING 5. Screen dump.

```

10 REM *****
20 REM |
30 REM | Amstrad Graphics Screen Dump |
40 REM |
50 REM | (C) Copyright Gary Nugent, Feb. 1985 |
60 REM |
70 REM *****
80 REM
90 ON BREAK GOSUB 1000
100 MODE 2:INK 0,2:INK 1,26:BORDER 11
110 REM
120 REM *** Set Top of Memory, Load Commands and Initialise RSX's ***
130 REM
140 MEMORY $41FF:LOAD"commands":CALL &AA0A
150 MOVE 200,385:DRAWR 223,0,1:LOCATE 26,2:PRINT CHR$(24)" AMSTRAD EXTENSION COM
MANDS "CHR$(24)
160 PRINT:PRINT" There follows an index of currently implemented Resident System
Extensions."
170 PRINT" and a brief description of their purpose."
180 MOVE 4,297:DRAWR 630,0,1:DRAWR 0,-291,1:DRAWR -630,0,1:DRAWR 0,291,1
190 MOVER 4,-4:DRAWR 622,0,1:DRAWR 0,-283,1:DRAWR -622,0,1:DRAWR 0,283,1
200 REM
210 REM *** List of Extension Commands ***
220 REM
230 LOCATE 4,8:PRINT CHR$(24)" ISCRDUMP "CHR$(24):LOCATE 40,8:PRINT"Graphics Scr
een Dump to Printer"
240 GOTO 240
250 BORDER 0:INK 0,0:INK 1,24:MODE 1:NEW:RETURN
1000 BORDER 0:INK 0,0:INK 1,24:MODE 1:NEW:RETURN

```

DIY PLOTTER

J. Pilkington continues his description of a low-cost, high performance X-Y plotter.

The design brief of the *E&CM* Plotter, outlined last month, was to produce a design combining low cost with high performance. To achieve this aim – a design that would not stretch the pockets of potential constructors – the plotter was based around low cost building materials: 'Swish' curtain track provides the basic framework of the mechanism.

In last month's article the basic construction was illustrated with photographs of the author's prototype. These details are supplemented this month by a series of drawings detailing various aspects of the design. These are not intended as definitive engineering drawings, but rather show the way in which the author tackled various aspects of implementing the design of the low cost plotter.

The information presented in this article, together with that to be found in Part 1 should allow the reasonably competent DIYer to complete the mechanics of the plotter. Next month the design of the stepper motor interface will be presented together with some examples of software for use with the design.

About the author – John Pilkington is Head of the Faculty of Design and Technology at St. Andrews Church of England School in Croydon. He will be pleased to discuss readers' ideas for improving his original concept to help any constructors facing any difficulties. He can be contacted via the Editor.

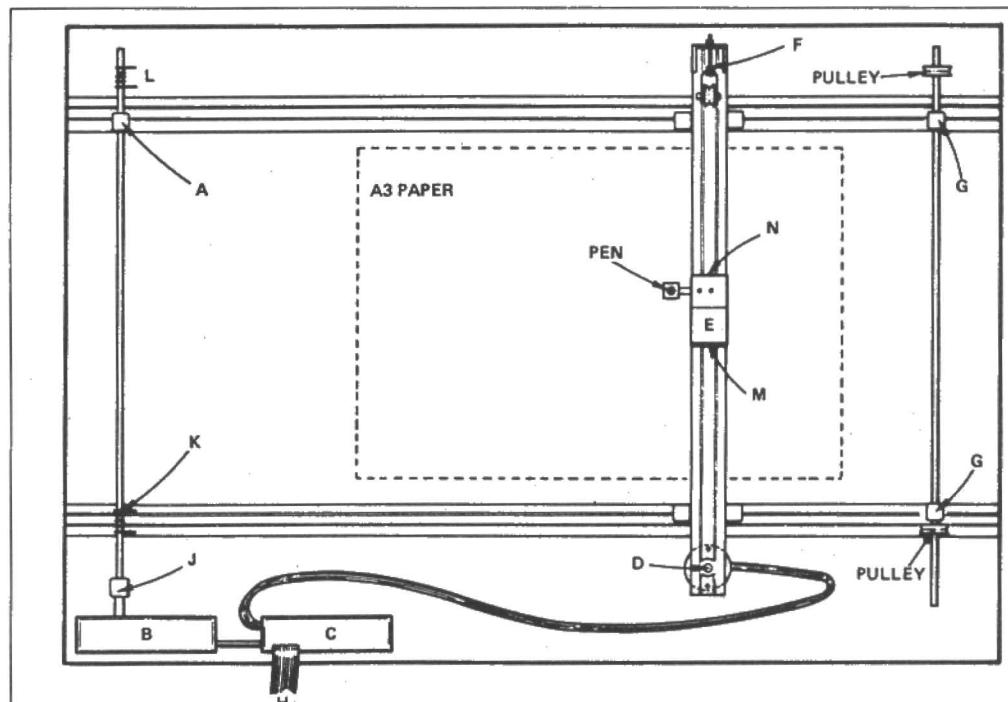
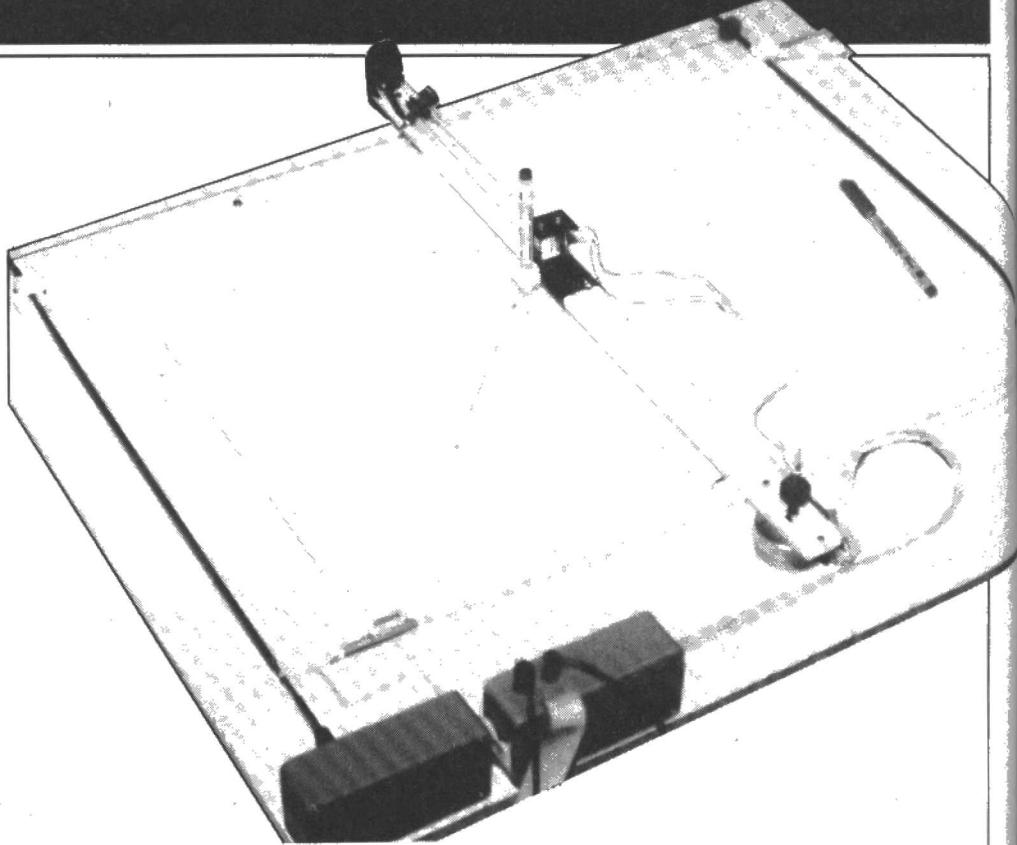
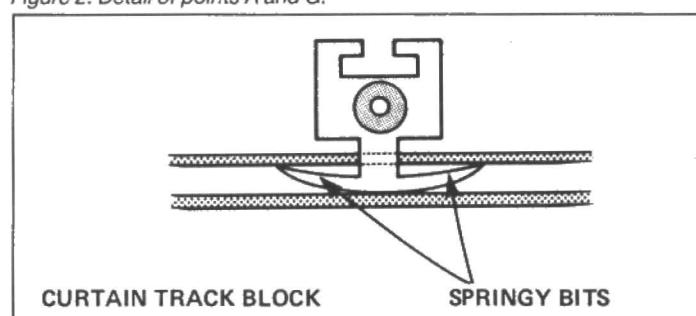


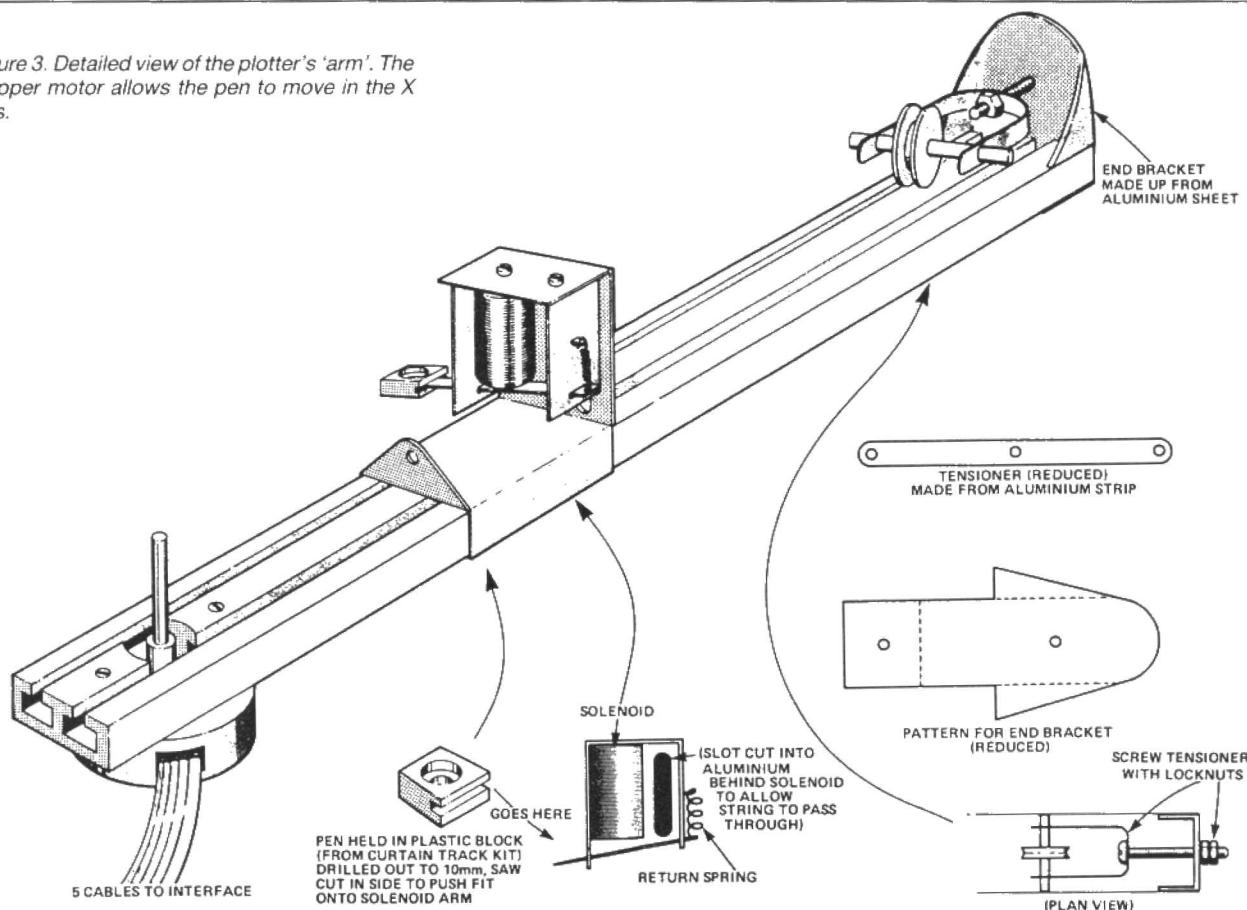
Figure 1. Plan view of the major components of the design.
Figure 2. Detail of points A and G.



To stack one on top of the other (points A and G in Figure 1) it is necessary to cut the 'springy' bits off the curtain track. Slide the base of A (Figure 1) into the top of B keeping the 'springy' parts on B.

A) Support for end of rod. Made by stacking two curtain tracks (see Figure 2). **B)** Box containing stepper motor 1 (x axis). **C)** Box containing stepper motor interfaces. **D)** Stepper motor 2 (y axis). **E)** Carriage holding pen and solenoid to lift pen. **G)** Blocks stacked to form support for rod (see Figure 2). **H)** 20 way ribbon cable from interface box to BBC micro. **J)** Small collar to convert stepper motor shaft diameter (4mm) to 3mm to give 0.2mm per step movement at the pen.

Figure 3. Detailed view of the plotter's 'arm'. The stepper motor allows the pen to move in the X axis.



NEW HOME COMPUTER CABINET

- ★ IT'S A DESK!
- ★ IT'S A STORAGE CABINET!
- ★ IT'S UNIQUE!
- ★ IT'S ONLY **£59.95**
incl. VAT + £6.00 p&p

Size:
24" W x 18" D
x 33 3/4" H



TEAK EFFECT OR EMBOSSED WHITE FINISH

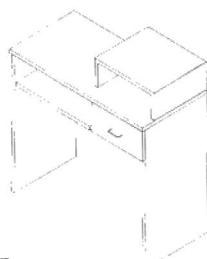
HOME COMPUTER DESK NEW DESIGN

- ★ PULL OUT DESK ON STEEL RUNNERS
- ★ FULL SIZE DRAWER
- ★ DETACHABLE MONITOR STAND
- ★ SPECIAL OFFER PRICE

£39.95 incl VAT + £6 p&p

Allow 28 days for delivery

Money Back Guarantee if not entirely satisfied
Send Cheque/PO/Access No. to:



CHALTON TIMBER PRODUCTS LTD
(Dept EC6) 139 Wigan Road, Euxton, Chorley, Lancs.

TELEPHONE: CHORLEY (02572) 74374

Sole Manufacturers. Available only from us (Trade Enquiries)

WD Software

For the QL:

WD Utilities (3rd ed)

(base £5.50)

PRINT 60-file DIRectory or view it on one screen, one-key LOAD, COPY or PRINT 60 files with one key (allows for namesakes). Multiple FORMATing to prevent corruption by stretching of tape. TOOLkit to give dated, numbered modules in program development. PRUNE old files to release space (one key DELETes a file). Full instructions in QUILL file. Use up to 6 EXTRA MICRODRIVES (add on your Spectrum ones).

WD Utilities for CST Disks

(base £8)

100-file capacity, for CST/Computamate disk system AND up to 4 extra microdrives. User-friendly timesavers.

RefQL (2nd ed)

(base £2)

300 useful QL references in an ARCHIVE file.

For Spectrum/QL/BBC:-

WD Morse Tutor

(base £4)

From absolute beginner to beyond RYA and Amateur Radio receiving. Adjust pitch. Set speed to your test level (4-18 wpm). Learn from single characters, via groups with wide spaces to random sentences; decrease spacing to normal. Write down what you hear, then CHECK on Screen or Printer (or speech for Spectrum fitted with Currah Micro-speech). Also own message, random figures, letters or mixed.

For Spectrum 48K:-

Tradewind

(base £4)

Sailing/trading strategy game with graphic surprises.

Jersey Quest

(base £4)

Text adventure with Bergerac and the Dragon. (not disk).

PRICES (incl. Europe postage - elsewhere add £1)

Spectrum/BBC Cassettes - base price only. QL or Spectrum Microdrives - £2/cartridge plus base price 5.25" floppies £2 plus base (SPDOS) format for Spectrum)

Two or more programs on one medium - pay medium + base. E.g. WD Utilities and RefQL for £9.50, but IMPOSSIBLE to mix QL/BBC/Spectrum programs on one medium. Send YOUR cartridge and base price, but FORMAT it FIRST in your DRIVE 1 for compatibility.

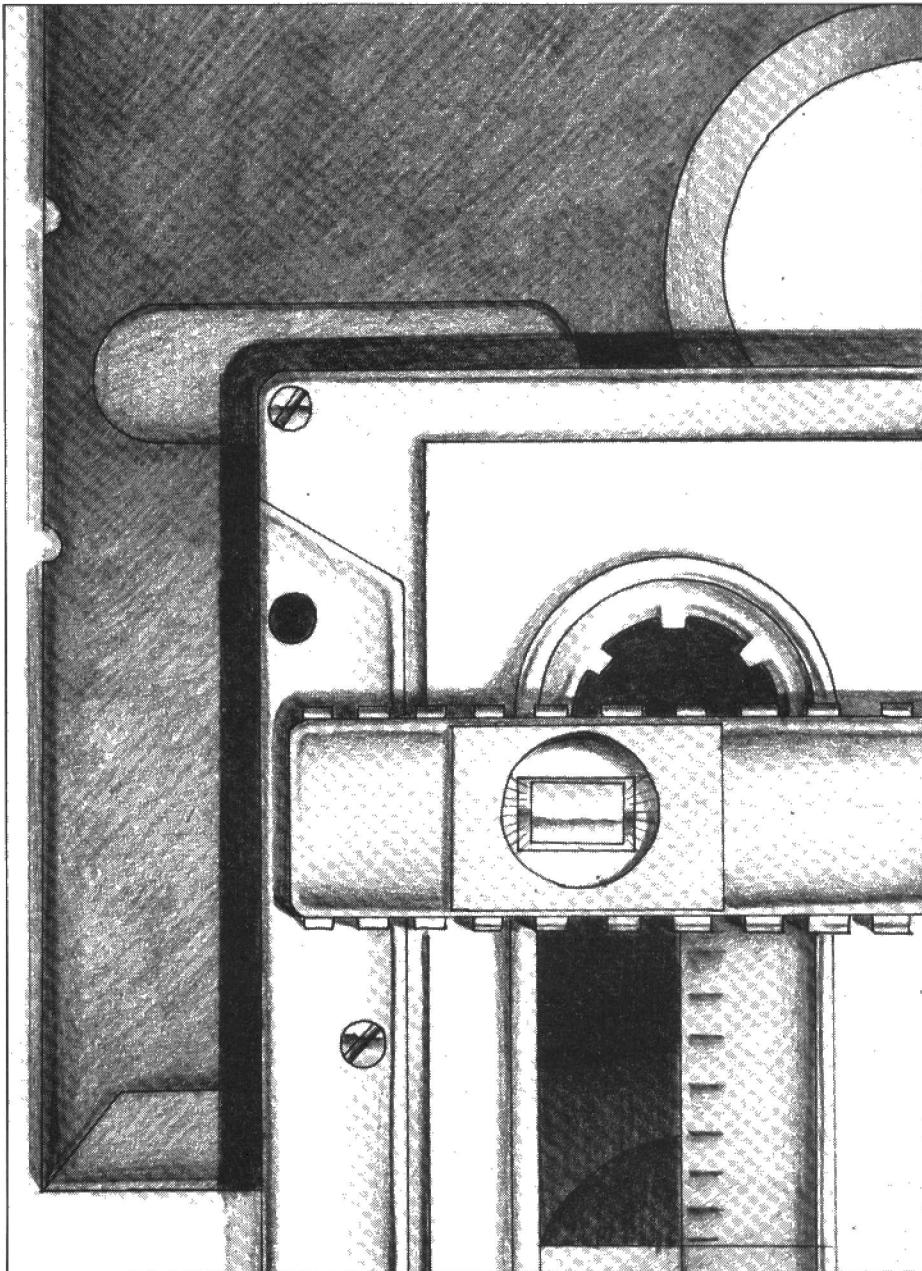
WDSoftware,
Hilltop, St Mary, Jersey. Tel: (0534) 81392

Making a mark

A new mega-series on the theory and practice of storing bits, bytes and megabytes, by Mike James.

In this series the different methods of mass storage will be described and we will explore how the familiar storage devices – tape and disk – work. The emphasis will fall not on pure theory but on how common problems can be overcome using a little understanding. So if you have always wondered what goes on inside a tape or disk drive then don't miss future installments! As well as looking at how things work from the hardware point of view, the way that software (that is operating systems) uses the different devices will also be covered. In other words, we will discover some of the mysteries of disk formats and why machine A can't (or won't) read a disk produced by machine B. Of course the object of knowing why machines use incompatible disk formats is to find a way around the incompatibility. Sometimes this is possible and even very easy: at other times it is better to know you are beaten and look for a different way of exchanging data!

Finally, a series on storage would be incomplete without at least a brief look at what the future might offer. In this case it looks as though the future might not be too far away. As always in computing, each of the component technologies is on the verge of a revolution and for mass storage the most talked about breakthrough is the use of optical methods to achieve vast storage potential at low cost. However, while all the frantic research and product development is going into optical disks etc, some remarkable but less revolutionary innovations are being incorporated into some of the familiar magnetic storage devices that will make them almost as good.



We all use different computers and different programs but the one thing that unites us is the need to store data in a form that can be archived for later use. So called 'mass storage' is the computer technology that causes most frustration among users – 'My program won't load', 'I've run out of space for data storage', 'Why can't machine x read data from machine y?' – and yet it is a comparatively simple technology! (In principle at least).

Not only does storage hardware frus-

trate individual users, but it is currently holding back the application of computers to many of the tasks that we all assume are right for computerisation. For example, why do libraries, encyclopedias, printed telephone directories etc. still exist? Although there are other factors such as the availability of suitable communications lines and the problems of transferring existing print to more up-to-date storage media, the plain fact is that paper is still cheaper than computer mass storage.



Primary and Secondary storage

What distinguishes mass storage from any other type of computer memory? Any computer is composed of two main parts: the CPU, which is responsible for obeying the instructions that make up your programs, and the memory, which is responsible for storing the program that the CPU is currently obeying. Now, while the memory that the CPU uses to read and write data comes in a number of different forms, RAM, ROM, EPROM etc, they all share one characteristic – the CPU can access any location very quickly. This type of memory is usually called 'primary' or 'main' memory and in the jargon it is said to be 'fast and intimately connected' to the CPU. Typical access times for primary memory are of the order of microseconds.

Another less obvious feature of primary memory is that the CPU can gain immediate access to the smallest unit of data that the machine can process in one operation. In eight bit machines this corresponds to retrieving or storing a single byte at a time.

In principle there is really no reason to use any type of storage other than primary memory. If the primary memory was big enough it could hold all the programs and data that the machine would ever want to use. However, in any computer system primary memory is always made as fast as the current technology will allow and this implies that primary memory is always going to be more expensive per byte than other, slower types of storage. Also, although it is possible to design ways of using primary memory as a method of exchanging data (ROM packs for example) it is inconvenient because of the close connection between it and the CPU. In practice, as well as primary memory, a computer system needs memory that is cheap, permanent, exchangeable and offers large amounts of storage space. In addition this type of memory doesn't have to be the fastest available – although the faster the better – and it doesn't have to be so closely connected to the CPU. Such memory is generally referred to as 'secondary storage' but it goes under many other names such as 'removable storage', 'mass storage', 'backup storage' etc.

Addressing

The idea that secondary storage is generally slower than primary storage is easy to understand, but you might be a little worried about how it can be less intimately connected to the CPU. Primary storage is divided up into locations, each of which can store the smallest unit of data that the

CPU works with in one operation. These days the most common unit of storage is the 'eight-bit byte' or just 'byte' for short. One byte represents eight bits of storage which in practical terms can be thought of as roughly one character or two digits. Which location the CPU wants to use is determined by its address within the memory.

To address two memory locations takes only a single bit, or in hardware terms, one address line (ie memory location 0 and memory location 1). To address four memory locations takes two bits (ie location 00, 01 and 11) and hence two address lines and so on. Each time we add one bit to the address and an address line to the hardware the amount of memory that the CPU can use doubles. Now it should be obvious why the primary memory size of most personal computers goes up in such strange increments. Most of the eight-bit computers based on the 6502, Z80, 6809 etc were equipped with 16 address lines and if you keep on doubling the amount of memory in the way described you will find that 16 address lines can access 65,536 memory locations. The number 65,536 seems a lot more familiar (and memorable) when written as 64K and demonstrates the main reason why we measure storage in terms of 1K-1024 locations. It is also worth knowing that 1K is the number of memory locations that you can access using ten bits.

You might think that the most obvious way to extend the amount of storage that the CPU can access is to increase the number of bits in the address. The trouble with this simple approach is that each bit that you add to the address increases the work that the CPU has to do (to calculate the address of the next memory location it requires). A further problem is that each address line from the CPU to the memory increases the complexity of the hardware. In practice it is easier to compromise: by keeping the size of the primary memory to 64K for eight-bit machines and to anything up to 64Mbytes for 16 and 32-bit machines.

Serial and block storage

It is the use of an address for each memory location that results in primary memory being closely connected to the CPU. In the case of secondary storage the use of an address for each item of data is avoided in either of two ways.

Firstly, the data can be manipulated one item at a time and simply stored in order. This is usually referred to as 'serial storage'. For example when using tape, items of data are written out one at a time and are stored in order on the tape. Reading the data back is simply a matter of rewinding the tape and reading the items back in the same order that they were written out. When using serial storage the CPU doesn't have to specify the exact address of an item of data stored on the tape. Indeed the only things that the CPU can do is to "write the next item", "read the next item" and

"rewind". Serial devices are simple but limited when it comes to altering the order of lists of data.

The second method is to use 'block storage'. Instead of giving each storage location an address, block storage involves giving collections of storage locations individual addresses. For example a disk drive is a block storage device and data is stored and retrieved in units of anything from 128 to 1024 bytes at a time. In the case of a disk such a collection of storage locations is referred to as a 'sector', but more of this later in the series. The point is that the CPU can give a command to the disk drive that amounts to "read the data in sector 56" (or whatever sector number is required) and each sector address identifies a block of data, not just a single item. Using a 16-bit address the CPU can access 64Kbytes of primary storage but using the same number of bits it can access something like 16Mbytes of secondary block storage (assuming a block size of 256 bytes which is very common).

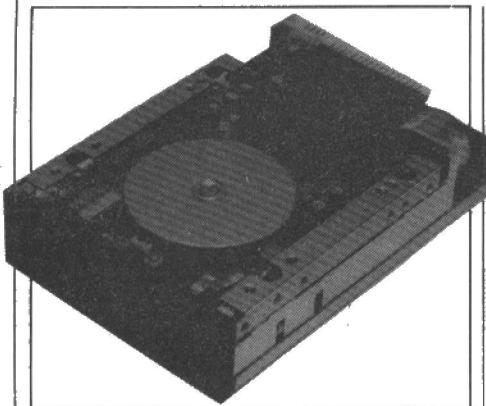
You can see that block storage gives the CPU access to large quantities of data with only a modest number of address bits. To a certain extent it is also possible for the CPU to gain access to the data in any order and hence such block devices are also called 'random access' devices. Block devices are ideal for storing data that naturally occurs in chunks such as information records, but with the help of a little software they can be used for almost any type of storage. There is a great deal more to be said about block devices and how they work but this is better left to a discussion of the archetypal block device – the disk drive – later in the series.

Principles of storage

Surprisingly there are only three ways of storing information (I would be interested to hear of any other candidates)

- Mechanical changes, eg physical deformation or overlaying of some base material
- Magnetisation
- Electrostatic charge storage

At first it is difficult to believe that all of the information storage devices that we use fit into one of these three categories, but when you look into the how each device works you will find that they are more or less sophisticated applications of these principles. For example, pencil and paper is an early type of mechanical data recording device, somewhat more up-to-date is the punched card or paper tape where holes are used to record individual bits. What you might not realise is that the storage medium of tomorrow, optical disk, is in fact a direct relation of the punch card – that is it is a mechanical storage device. Data is recorded on an optical disk by using a laser to burn small pits or blisters on the surface and these are directly analogous to the holes in a punched card – so who says optical disks are new technology!



Magnetism

Of the three methods of storing data, magnetic recording is the best known and most widely used. Not only does it turn up in computers in the form of tape and disk but it is the basis of both audio and video recording. The underlying principle behind all magnetic recording is simply that some, initially unmagnetised materials, can be magnetised by being subjected to a magnetic field. The rest of the story is well known to most people even slightly interested in electronics – a current flowing in a coil can create a magnetic field which can be used to alter the magnetisation of a

Although magnetic recording has been with us for a long time it is still a very attractive method both in terms of cost and performance. Current research work promises to increase the storage capacity of traditional magnetic devices by an order of magnitude – making them directly comparable with newer technologies. Speed of access is also being improved and even though all magnetic devices (other than bubble memory) are limited by the movement of mechanical parts there is still much room for improvement. In the medium term, and perhaps the long term future, magnetism still has its attractions.

the only sort of secondary storage that makes any sense has to be non-volatile. There are ways of making standard RAM chips non-volatile by adding small rechargeable batteries etc but this is usually thought to be impractical for all except small areas of storage (tens of bytes) for very important data.

It is possible to build chips that will retain patterns of small charges stored in capacitors for tens of years, indeed the familiar EPROM works in exactly this way, and this means that it is possible to imagine secondary storage based on the familiar silicon technology used in the rest of the computer. Indeed Sinclair Research is promising enormous single chips (wafers) with storage capacities in the 1Mbyte range which are intended to be used as fast secondary storage for the QL. Such chips are so new that there isn't a generally accepted jargon word for them but 'silicon disks' seems like a good bet. It could be that the future of secondary storage lies with the silicon wafer in the same way that the future of the rest of computing lies in the silicon chip.

As well as using charge storage within memory chips, it is possible to think up ways of using charge to record data in the same way that magnetism is used. For example I have seen the electrostatic equivalent of a magnetic tape recorder. It used highly insulating tape and 'sprayed' a high voltage (several kilo volts!) as the tape moved passed, using a recording head that was nothing more than a pair of sharp edges! Needless to say the idea never caught on and was really only built as a curiosity but it did prove that to every magnetic device there is an equivalent, if not always practical, electrostatic device.

Before moving on to consider the details of particular storage devices it might be helpful to give some idea of the amount of storage for current storage devices and their typical access times. See Table 1.

"Surprisingly there are only three ways of storing information – any storage device involves more or less sophisticated applications of a few basic principles".

portion of a tape or a disk: the data can be read by moving another coil over the magnetic variations which induce a current to flow in the coil proportional to the magnetisation. This simple idea of converting a varying current into a pattern of magnetisation and vice versa is more difficult to turn into a working, reliable storage device than you might think; some of these difficulties will be described in the articles that describe how tape and disk storage work.

There are many ways of using the magnetic recording principle to create a storage device, and the family of magnetic storage devices is bewilderingly large. As well as many tape formats, recording methods, disk sizes, disk types, magnetic strip cards, magnetic character printers and readers there is also an apparently separate family of devices – bubble memory. Bubble memory is a magnetic recording device with no mechanical moving parts! Essentially what happens is that the patterns of magnetic variation are created in a special type of crystal in such a way that they can be coaxed to move past a fixed reading head. In other words, the magnetic pattern moves in an orderly fashion while the material on which it is recorded remains static. Bubble memory is faster than magnetic devices that do involve moving parts but slower than RAM and so it sits in a slightly grey area between primary and secondary storage.

The trouble with electric charge is that, unlike magnetism it tends to leak away. Even the best insulators that we have are not good enough to stop the minute amounts of charge that are stored in a dynamic RAM capacitor from leaking away in less than a millisecond. As a result elec-

TABLE 1.

Device	Typical capacity	Speed
Audio tape	50K → 200K	30b/s → 120b/s
Digital tape	20M → 60M	30Kb/s → 90Kb/s
Floppy disk 8"	600K → 1.2M	250Kb/s → 500Kb/s
Floppy disk 5"	80K → 1.6M	125Kb/s → 500Kb/s
Floppy disk 3"	80K → 1M	125Kb/s → 250Kb/s
Hard disk 8"	50M → 200M	600Kb/s → 1Mb/s
Hard disk 5"	10M → 170M	600Kb/s →
Hard disk 3"	5M → 10M	
Optical disk	300M → 2G	1Mb/s → 2Mb/s

trostatic devices need constant attention to make sure that they don't 'forget' the data stored in them. In the case of dynamic RAMs this takes the form of 'refresh' cycles that replenish the charge in the capacitors, in the case of static RAM it takes the form of a constant flow of current holding a flip-flop in a particular state. This need for constant attention is very noticeable if you turn the power off when, of course, nearly all RAMs lose whatever was stored in them. Such devices are usually referred to as 'volatile' and while volatile storage devices are OK for primary storage

The transfer speeds are all in terms of Kbytes per second and 1G ('G' stands for Gigabyte) is 1000 Mbytes. Notice that if you need a particular storage capacity coupled with a particular access speed then there is usually more than one device to choose from; which one is best depends on factors such as cost, size, robustness, and whether the recording media is removable or even reusable.

Next month Tape recorders, how they work and how to make them a reliable method of data storage.

QSHELL

QShell, by Adam Denning, is a major software development which over the next four issues will turn into an MSDOS type front-end for QDOS, the QL operating system.

Although the SuperBasic interpreter does an excellent job as the QDOS command line interpreter, there are occasions when an operating system more like those found on 8088 and 8086-based machines would be useful.

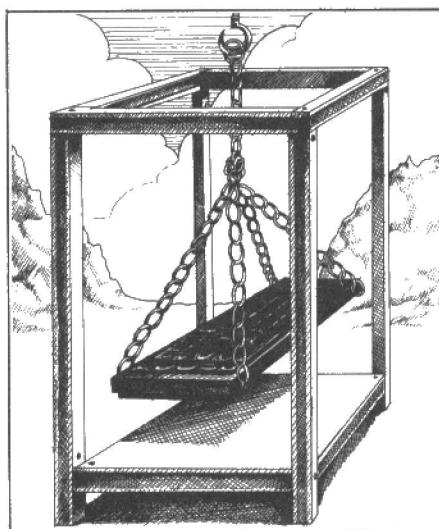
The extensive program which will be developed over the next few issues goes some way towards filling this need. It has wildcard facilities for file copying, deletion, renaming and so on, it has a rudimentary 'shell' language and a batch file capability, and of course it provides easy access to QDOS features such as multi-tasking, job control and device-independent input and output.

As the source code is presented, each part will be explained, so alterations and additions will be easy for those who wish to customise the system - like most applications and systems, it will benefit from the addition of memory to the basic machine, but it is kept as small as possible by being written entirely in 68000 assembly language. A couple of macros for invoking QDOS trap routines and vectored utilities are used extensively, but if the assembler you're using doesn't support macros then simply substitute each macro invocation with the relevant group of instructions. The two macros are shown in Listing 1.

As an enormous number of QDOS routines are used, the definitions for various system constants are kept in a header file which I call header.asm. If you do not have your own file, you are strongly advised to buy the QL Technical Manual available from Sinclair Research.

Once the program is invoked (we'll call it the shell from now on), it takes over from the SuperBasic interpreter using the machine code equivalent of EXEC_W. This means that the interpreter is unavailable, so the program is unlikely to be of interest to those committed to programming in Basic! (I know of none).

To facilitate this invocation, a short machine code addition is made to SuperBasic which manifests itself as a procedure called SHELL. When this procedure is called, the shell is invoked by loading it from disk. In the same file as the shell procedure you will see that there is an updated



LISTING 1. Two macros.

JUMP	EQU	0	constants for macro expansion
CALL	EQU	1	
QDOS	MACRO		A general-purpose QDOS macro
	MOVE.B	#\1,00	
	TRAP	#\2	
	ENDM		
VECTOR	MACRO		A general-purpose QDOS utility vector macro
	MOVE.W	\1,\2	
	IFNE	\3	
	JSR	(\2)	
	MEXIT		
	ENDC		
	JMP	(\2)	
	ENDM		

version of the function key program, with the bugs removed. This version uses the common heap space rather than the resident procedure area to store the string definitions. If you don't want to use the shell, then this listing by itself provides a very useful extension to the system.

SHELL is deceptively simple. It first attempts to open a file called flp1_init_code, which contains the shell program. Obviously any name and device can be used. If this file cannot be opened the pro-

cedure reports back to Basic and ends.

The file's header is read in to see how long it and its data space are, and then a job is created using these parameters and MT_CJOB. If this creation fails for any reason, the error is reported and the procedure returns. Otherwise the file is loaded into the space allocated by QDOS and subsequently closed. Using a method identical to that used in the article on pipes (see *E&CM* March 1985), the program then plays around with the shell's stack. It looks in the Basic channel tables and saves the channel IDs for channels 0, 1 and 2, and then it saves the address of the table of pointers used for function key definition. Finally, when the four parameters have passed to the shell, a word of 4 is put on the stack to tell it so. The reason that the channel IDs are passed to shell rather than opening up its own channels is that it saves a (tiny) bit of memory.

'QShell is written entirely in 68000 assembly language'.

The shell job is then activated with a priority of 16 and a timeout of -1, which means that the SuperBasic interpreter will be suspended from there on. It will only be released when shell is finished, which occurs when the BASIC command is issued. We'll be looking at that later.

This file should be assembled and the length of the resultant code noted. The procedures can then be added to the interpreter by reserving resident procedure

area. Suppose the file was 800 bytes long, for example. It could be loaded like this:

```
a=respr(800)
1bytes flp1_init_code,a
call a
new
```

Once the code has been called, SHELL and KEY are added to the SuperBasic name table, and the function key routines is linked into the 50Hz task list.

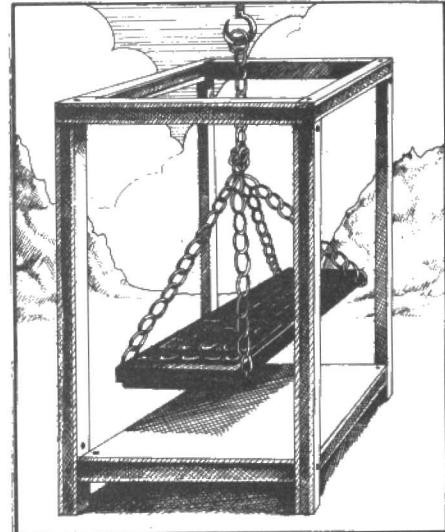
The shell also uses a clock, and although

LISTING 2. Initial procedures, user defined function keys, SHELL procedure.

* Initial procedures - a procedure to reset the machine and one to kill jobs
 + Adam Denning 22nd January 1985 (C) 1985 AD
 * User-defined function keys added 11th March 1985
 * COUNT function removed and SHELL procedure added 22nd March 1985
 * SHELL procedure altered to activate with a priority of 16 8th April 1985

GET	"f1p2_header_asa"	MOVEQ	0-1,D3	MOVE.B	2(A6,A1.L),D1
		QDOS	MT_ACTIV,1	SUBI.B	#'0',D1
START	LEA.L PROC_DEF,A1	BRA.S	EXIT_SHL	CMPI.W	\$1,D2
VECTOR	BP_INIT,2,CALL	EXIT_SHL2	MOVE.L	BEG.S	LONKEYN
LEA.L	KEY_LINK,A0		(A7)+,D1	MOVE.B	3(A6,A1.L),D3
LEA.L	KEY_INT,A1		MOVEM.L	SUBI.B	#'0',D3
MOVE.L	A1,4(A0)		AO/DO,-(A7)	BMI	EXIT_KEYS
QDOS	MT_LPOLL,1		QDOS	MULU	\$10,D1
RTS		EXIT_SHL1	MT_FRJDB,1	ADD.B	D3,D1
PROC_DEF	DC.W 2	MOVE.L	MOVE.M	CMPI.B	\$1,D1
			(A7)+,AO/DO	BLT	EXIT_KEYS
DC.W	KEYS_PROC+*	KEY_INT	DO,-(A7)	CMPI.B	\$15,D1
DC.B	3	MOVE.L	QDOS	BGT.S	EXIT_KEYS
DC.B	'KEY'		IO_CLOSE,2	SUBB.B	\$1,D1
DC.W	SHEL_PROC+*	MOVE.L	MOVE.L	MOVE.W	\$6,D3
DC.B	5		(A7)+,D0	MULU	D3,D1
DC.B	'SHELL'	KEYCONT		ADDQ.W	\$2,D1
DC.W	0	SUBB.L	#1,A1	LEA.L	KEYTAB,A0
DC.W	0	MOVE.B	(A1),D1	ADDQ.L	D1,A0
DC.W	0	BEQ.S	ENDREPL	TST.W	D6
* The SHELL procedure					
		CMPI.B	\$#E2,D1	BEQ.S	KEYFULL
		BNE.S	KEYBROM	TST.L	(A0)
		MOVE.B	ENDOFQ(A2),A1	BEQ.S	EMPTYKEY
		MOVE.L		MOVE.L	(A0),A1
		KEYCONT	SUBB.L	SUBA.L	AO,A0
			#1,A1	VECTOR	UT_MTEXT,2,JUMP
			MOVE.B	ADDQ.W	\$1,D2
			(A1),D1	ANDI.W	\$-2,D2
			BEQ.S	ADDA.L	D2,A1
			ENDREPL	MOVE.M	AO-A1,-(A7)
				TST.L	(A0)
				BEQ.S	NOCHSP
				MOVE.L	(A0),AO
				QDOS	MT_RECV,1
				MOVE.L	(A7),AO
				MOVE.L	4(A7),A1
SHEL_PROC	LEA.L SHELLFILE,A0	KEYBROM	LEA.L FUNCSON,A1		
	MOVEQ	0-1,D1	TST.W (A1)		
	OPEN_INS,D3	BNE.S ENDREPL			
	IO_OPEN,2	LEA.L KEYTAB,A3			
	TST.L DO	LEA.L ENDOFTAB,A0			
	BNE EXIT_SHL	KEYLOOP CMP.B (A3),D1			
	MOVEQ	BEQ.S KEYCHANGE			
	#16,D2	ADDD.Q \$6,A3			
	MOVEQ	CMPA.L A3,A0			
	#1,D3	BNE.S KEYLOOP			
	LEA.L H_SPACE,A1	ENDREPL RTS			
	QDOS FS_HEADR,3	KEYCHANGE MOVE.L 2(A3),D4			
	MOVE.L A0,-(A7)	BEQ.S ENDREPL			
	LEA.L H_SPACE,A1	MOVEA.L D4,A4			
	MOVE.L (A1),D2	MOVEQ #0,D4			
	MOVE.L 6(A1),D3	MOVE.W (A4)+,D4			
	MOVEQ #1,D1	SUBB.W \$1,D4			
	SUBA.L A1,A1	BMI.S ENDREPL			
	QDOS MT_CDOB,1	MOVE.B (A4)+,D1			
	MOVE.L A0,A1	VECTOR IO_BIN,0,CALL			
	MOVE.L (A7)+,AO	TST.L DO			
	TST.L DO	BNE.S ENDREPL			
	BNE.S EXIT_SHL1	DBRA D4,SWAPKEY			
	MOVE.L A1,A2	BRA.S ENDREPL			
	MOVE.L D1,-(A7)				
	MOVEQ #1,D3				
	QDOS FS_LOAD,3				
	TST.L DO				
	BNE.S EXIT_SHL2				
	QDOS IO_CLOSE,2				
	BUBA.L 0(JOB_AREA-SAVE_USP),A2				
	MOVE.L (A2),A1				
	MOVE.L BV_CHBAS(A6),A3	KEYS_PROC VECTOR CA_BTSTR,2,CALL			
	MOVE.L 0(A6,A3.L),-(A1)	BNE EXIT_KEYS			
	ADDA.L #CH_LENGTH,A3	MOVEQ #ERR_BP,DO			
	MOVE.L 0(A6,A3.L),-(A1)	MOVEQ #0,D6			
	ADDA.L #CH_LENGTH,A3	CMPI.W #2,D3			
	MOVE.L 0(A6,A3.L),-(A1)	BEG.S FULLKEYS			
	LEA.L KEYTAB,A3	CMPI.W #1,D3			
	MOVE.L A3,-(A1)	BNE EXIT_KEYS			
	MOVE.W #4,-(A1)	MOVEQ #1,D6			
	MOVE.L A1,(A2)	MOVEQ #0,D1			
	MOVE.L (A7)+,D1	MOVEQ #0,D2			
	MOVEQ #16,D2	MOVE.W 0(A6,A1.L),B2			
				NOCHSP MOVEQ #0,D2	
				MOVE.L D2,D1	
				MOVE.L D1,(A0)	
				MOVE.W 2(A6,A1.L),D1	
				BNE.S NOTNULL	
				MOVE.M (A7)+,AO-A1	
				BRA.S EMPTYKEY	
				MOVE.W D1,D6	
				ADDD.W #1,D1	
				ANDI.W #2,D1	
				MOVE.L D1,-(A7)	
				ADDD.W #2,D1	

* A SuperBasic extension to program the function keys



PROJECT

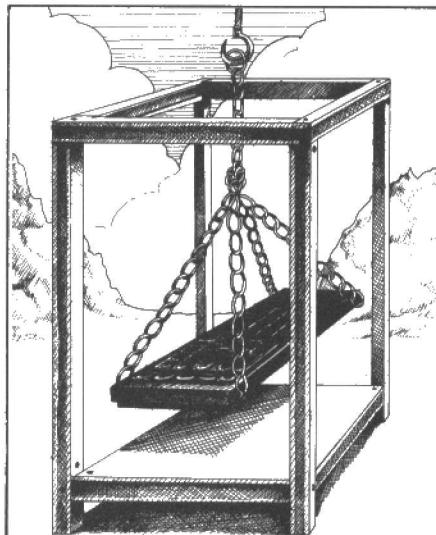
LISTING 2 (Continued)

QDOS	MT_ALCPNP,1		
MOVE.L	A0,A2	DC.L	0
MOVE.M	(A7)+,D1/A0-A1	DC.B	\$F4,0
TST.L	DO	DC.L	0
BNE.S	EXIT_KEYS	DC.B	\$FB,0
MOVE.L	A2,(A0)	DC.L	0
MOVE.W	D6,(A2)+	DC.B	\$EA,0
LSR.L	#1,D1	DC.L	0
SUBQ.W	#1,D1	DC.B	\$EE,0
MOVELOOP	MOVE.W 4(A6,A1,L),(A2)+	DC.L	0
	ADDQ.L #2,A1	DC.B	\$F2,0
	DBRA D1,MOVELOOP	DC.L	0
EMPTYKEY	MOVEQ #0,DO	DC.B	\$F6,0
EXIT_KEYS	RTS	DC.L	0
KEY_LINK	DS.L 2	DC.B	\$FA,0
FUNCSON	DC.W 0	DC.L	0
SHELLFILE	DC.W 10	DC.B	\$EB,0
	DC.B 'f1p1_shell'	DC.L	0
H_SPACE	DS.L 4	DC.B	\$EF,0
* A table of key codes			
KEYTAB	DC.B \$EB,0	F1	DC.L 0
	DC.L 0		DC.B \$FB,0
	DC.B \$EC,0	F2	DC.L 0
	DC.L 0		DC.B \$F3,0
	DC.B \$FO,0	F3	DC.L 0
		ENDOFTAB	END

any clock program will do, that in Listing 3 is recommended as it has the window in the right position and is as short as possible. Note that it does not set its own priority – this is done by the shell when it loads, creates and activates it. Once this file has been assembled, save it under the name of f1p1_sclock (or mdv1_sclock).

We'll start to write the code for shell itself next month, but we can take this opportunity to discuss just what we want it to do. As it has been passed three channel IDs, we can use these three screen windows for different purposes. Channel 0 is the command window, where all commands are typed and error messages received. Channel 1 can be used for general purposes, such as directory listings and dumped or typed files. Channel 2 will be a status window, telling us exactly what the system is up to.

When typing a command to the shell, it is useful to be able to use upper or lower case, and to edit the line in the normal way before finally pressing ENTER. If the command is not one of those built in, it seems a good idea to make it look for a file of that name and execute it as a job. This is similar to the action of 'external' MS-DOS commands such as LINK and FORMAT. To make things even easier, we'll adopt a 'default drive' system, so that we don't need to specify the drive for most operations.



better at multi-tasking than the IBM PC, so we would still be able to work while our documents are being printed!

You'll remember that the SuperBasic SHELL procedure also passes the address of the table of pointers to function key definitions, so naturally we will incorporate a facility to allow us to program the keys from the shell.

A command typed into a command line interpreter, which is essentially just what the shell program is, can be thought of as

'QShell provides easy access to QDOS procedures such as multitasking and job control'.

Other useful features would be a 'turn the printer on' facility, which would cause all output that is sent to channel 1 to be similarly echoed on our printer, and a printer spooler running as a background task. This latter would be rather like MS-DOS PRINT, except that the QL is rather

consisting of two parts; the command itself and a 'command tail' which is optional. This command tail, if present, would hold information to be passed to the command, such as the names of files to copy in a COPY command, or an expression to evaluate in a calculator. We will need to

write a couple of subroutines to separate command lines in this way, and to keep track of the pointers to each command line section.

Although we can easily write our own jobs which will accept command lines, a great deal of software has already been written for the QL and in general, these programs are not capable of accepting command line arguments so we can't use the shell program to pass parameters to them in this way. This means that although we could type

ed

to invoke a screen editor called 'ed', we couldn't type

ed myfile

to make it automatically load and commence editing 'myfile'. We would need to answer the program's prompts in the normal way.

The shell program falls naturally into three different sections:

THE INITIALISATION

setting up windows, default channels and so on

THE COMMAND LINE INTERPRETER

reading a command in from the keyboard and decoding it, reporting errors and calling the command routines

THE COMMAND ROUTINES

the set of built-in and external subroutines or programs which perform the actions required, such as DELETE, DIR and so on.

Certain commands can act upon a group of files in one invocation using wildcards. As this involves a tedious piece of code which is not really worth repeating again and again, we will encompass all these routines in a sort of 'wildcard subsystem'. COPY, RENAME and so on will simply pass their parameters to this subsystem, where the files matching the specification are found and acted upon in the desired way.

This may appear to mean that a structure is imposed on the filenames used within the shell, but in practice this doesn't have to be the case as we can allow one specification to cover many. However, it is sensible and more logical to limit filenames to just two parts – a filename and an extension:

filename_extension

so that when the device name is appended we can separate each constituent part by looking for the underscore character. However, by selecting a default directory rather than merely a device, we can implement the bare bones of a directory tree structure upon our filing system. We will discuss this further when we look at the USE command.

Next month we will write the initialisation section and look at ways of implementing the command line interpreter.

LISTING 3. Clock

* Executable clock program for the QL
* Special version for SHELL command interpreter
* By Adam Denning 22nd March 1985
* Copyright (C) 1985 Adam Denning

BET	"flp2_header_asn"		
SIZE	150		
BRA.S	C_START	Ignore standard format code	
DC.L	0		
DC.W	\$4AFB	Standard format identification	
DC.W	5	Program name	
DC.B	'CLOCK',0		
		PBLOCK	
C_START	LEA.L	PBLOCK,A1	Open screen window using the definitions
	VECTOR	UT_SCR,2,CALL	in PBLOCK
RE_ENTRY	MOVEQ	#0,D1	Set cursor position to 0,0
	MOVEQ	#0,D2	
	MOVEQ	#-1,D3	
	QDOS	SD_POS,3	
	MOVE.L	A0,-(A7)	Save channel ID
	QDOS	MT_RCLK,1	Read the clock into D1.L
	MOVE.L	(A7)+,A0	Retrieve channel ID
	MOVE.L	D1,-(A7)	Save the read time
	LEA.L	50(A4),A1	Put suitable stack address in A1
	VECTOR	CN_DATE,2,CALL	A4 holds offset from A6 of start of data

		area; convert date to
ADDA.L	A6,A1	ASCII
VECTOR	UT_MTEXT,2,CALL	UT_MTEXT needs A1 pointing to
MOVEQ	#' ',D1	an absolute address, not A6 relative
QDOS	IO_SBYTE,3	Send a space
MOVE.L	(A7)+,D1	Infinite timeout
LEA.L	50(A4),A1	
VECTOR	CN_DAY,2,CALL	Retrieve clock time
ADDA.L	A6,A1	Suitable stack address in A1
VECTOR	UT_MTEXT,2,CALL	Convert to ASCII day string
PBLOCK	RE_ENTRY	Change from A6 rel to absolute
DC.W	\$201	Print day
DC.W	\$400	Loop as scheduler permits
DC.W	148	Red one-pixel border
DC.W	12	Green paper black ink
DC.W	364	Window width
DC.W	244	Window height
END		Window X position
END		Window Y position

E&CM PCB SERVICE

October 1983		
BBC EPROM Programmer	£6.66
December 1983		
BBC Sideways RAM	£6.48
February 1984		
BBC Sideways ROM Board	£7.13
March 1984		
Spectrum Cassette Controller	£2.59
April 1984		
Commodore A/D	£2.15
May 1984		
Memex	£7.55
Spectrum Diary	£4.26
Centronics Buffer	£7.41
June 1984		
Mains Data Link (2 Boards)	£4.72
July 1984		
IR Data Link (2 Boards)	£3.95
August 1984		
Robot Wall Builder	£2.70
September 1984		
Spectrum Frequency Meter	£3.61
October 1984		
EPROM Simulator	£5.85
November 1984		
Amstrad PIO	£5.65

HOW TO ORDER

HOW TO ORDER
List the boards required and add 50p post and packing charge to the total cost of the boards. Send your order with a cheque or postal order to:

**E&CM PCB Service, Priory Court,
30-32 Farringdon Lane, London EC1R 3AU
Telephone: 01-251 6222**

Please supply the following PCBs:

Post & Packing 50p

TOTAL £

Date

Name (please print)

Address _____

Address: www.ox.ac.uk | www.ox.ac.uk/departmental/centres/centres_in_oxford/centres_in_oxford.html

PLEASE ALLOW 28 DAYS FOR DELIVERY

Eurohard means business

Eurohard took over responsibility for the Dragon computer last year, since when little has been heard of their plans. The company's development manager, Carlos Sicilia, visited the recent 6809 User's show and gave '68' computing an exclusive interview breaking this silence. He outlined plans for future developments of the Dragon itself and of other models of computer the company is to launch in the near future.

The first thing to emerge from our interview with Snr Sicilia was that the company had all but finished work on an MSX computer. Indeed he thought it quite possible that such a machine could be on sale in this country within the next few months. This fact gives a clue to the marketing policy that Eurohard has decided to adopt. The company plans to market the MSX as a games machine offering an entry point computer while the Dragon will be promoted as a machine for the 'serious' home user and as a low cost business machine.

The fact that the Dragon is to be launched into the business market explains a number of the refinements that Eurohard have made to the existing machine. The first is merely

cosmetic: the computer is to be repackaged in a slim line white case for a more up-market image than the present 'chunky' casing. The second development that has occupied much of Eurohard's R&D time over the past few months will not have any immediate benefits to users of the computer in this country, as it concerns the provision on the keyboard of the plethora of accents that need to be accessible if working in the majority of Eurohard languages.

Users of the Dragon will be far more interested in some of the developments that are still in the labs. The first of these concerns the provision of a true 80 column display capability. Motorola has been working on a replacement for the 6847 VDG for some time now but has still failed to supply even sample quantities of the device. At one stage it was hoped that any 6847 replacement would be pin compatible with the current device. The preliminary specifications for the new IC indicate that this is not to be the case and incorporating the 80 column version of the chip in future computers will mean some changes to the present PCB design. Eurohard have undertaken a redesign of the

board and at present are just waiting for sample ICs to plug into these engineering prototypes to confirm correct operation of the system.

Other plans include expanding the present 64K memory limit to 256K, or even 512K specification. To complement such memory expansion Eurohard are working hard on a disk interface.

These enhancements to the basic machine mean that it will offer the performance level of a machine suitable for use in a small business environment – the obvious question is what about the software? Seasoned 6809 users will know that there is little problem in this area as both FLEX and OS9 have a wealth of software available to run under them – everything from spreadsheets to wordprocessors taking in C compilers en route. There is one potential snag however, namely that any modifications to the hardware of the Dragon will require alterations to the OS in order that the system will operate correctly.

Eurohard have been talking to Microware, the author's of OS9 about implementing such changes to their source code but to date have found things

by Ken Alexander

rather slow going. When it comes to FLEX however Compusense have an arrangement with TSC (owners of the FLEX copyrights) that allows them to offer customised versions of the OS to companies throughout Europe. This fact will tip the balance in favour of FLEX as the OS under which future software for the Dragon will be written. An additional factor in favour of FLEX is that many people would consider this system more suited to the Dragon's specification. The code is far more compact and many of the frills of OS9 tend to make it more of an academic language rather than a workman-like commercial operating system.

It was obvious that Eurohard has given considerable thought to the way in which it will market the Dragon and MSX computers both in the UK and throughout Europe. An expanded Dragon with 256K of memory, the FLEX operating system with a hard disk interface available as an option could well give the likes of the QL, and the Atari ST computers, if and when they arrive, a good run for their money.

OS9 command separators

One of the major attractions of the OS9 operating system on 'large' 6809 systems is the fact that it offers both a multi-tasking and multi-user environment. But when used with a computer such as the Dragon these features turn out to be more of a liability than an asset. This is because the shortcomings in the hardware of the computer make it very difficult to conceive of occasions when the power of either multi-tasking or multi-user operation could be sensibly realised.

Having said that, it is worth briefly looking at the concurrent execution command separator and, for completeness, at the closely related sequential separator. The third valid OS9 command separator, used to construct pipelines, will be covered in a future article.

The sequential command

separator character is the ';' and using this character to separate each <program name> <parameter> from the next on a single program line will cause each command group to be executed in sequence.

An example of such a command line would be

```
COPY OLDSFILE NEWFILE ;
DEL OLDFILE ; LIST
NEWFILE
```

Note that the action initiated by this command line would be equivalent to that which would result if each command and associated parameter had been entered as a different line. All the programs in the line are in fact separate; 'child' processes of the OS9 shell. After initiating processes of a program to be executed sequentially, 'shell' enters a 'wait' state until

execution of the called program terminates.

& concurrent execution

The concurrent separator is the '&' which will cause a program to be run as a separate process but in this case the 'shell' will not wait for it to complete before processing the next command.

The Dragon OS9 manual gives as an example the following:

OS9: DIR>/p& The standard OS9 prompt followed by a print directory command and the '&' concurrent separator.

&007 Process ID number

OS9: OS9 prompt indicating that the system is awaiting a further command.

As indicated above this facility looks good on paper but in the case of the Dragon is of

little practical use. The above example serves to illustrate the point. The DIR command will cause the system to initiate disk activity in order to read the disk's directory. In the case of the Dragon, disk access virtually brings the rest of the system to a halt, suspending the interrupts that control the time slicing that in turn controls the multi-tasking process.

Thus while the screen may indicate that the system is ready to accept another command, typing anything at the keyboard is not likely to produce much of a result. The bottlenecks caused by disk access and the limited amount of memory available to OS9 in the Dragon, mean that the OS9 multi-tasking facility is about as much good as a 'go-faster' stripe painted down the side of a car.

Anatomy of the Dragon

Mike James describes how each component of the Dragon's hardware-overall, a model of elegant economy – slots into place.

The Dragon is a fascinating machine from the hardware point of view. It is almost entirely based on chips made by Motorola, and like large pieces of a jigsaw puzzle, each chip determines how the whole machine will look once they are put together. In this sense the Dragon represents a very standard approach to microcomputer design and there are no unexpected or clever uses of chips. However this isn't a criticism; the Dragon is a model of elegant economy in the use of LSI (large scale integration) to produce a versatile microcomputer.

In this article the way that the Dragon works is described by reference to its circuit diagram. This is a more detailed description than usually encountered in magazines and books, but it cannot claim to be the complete story; the Dragon is a careful mix of hardware and software and to fully understand it you need both sides of the coin.

An overview

The Dragon is composed of five large (40 pin) chips, one 6809 CPU (IC38), one SAM memory controller (IC39), one 6845 video controller (IC10) and two 6821 parallel interfaces (ICs 5 & 8). The CPU, SAM, and video controller work together to produce the central part of the machine using RAM and ROM to run programs and generate the memory mapped video display. The two 6821 PIAs are responsible for the remainder of the Dragon's I/O – the keyboard, the joysticks, sound generator and the cassette interface. The Dragon comes in two versions, the older Dragon 32 and the newer Dragon 64. The Dragon 32 has standard 4116 dynamic RAM chips which provide 32K of RAM. The Dragon 64 is essentially the same but uses 64K dynamic RAM chips that can be set to provide either 32K or a full 64K of RAM. In addition the Dragon 64 has a serial port provided by a 6522 Asynchronous Interface Adaptor (the ACIA – IC11). In the rest of this article the Dragon 64 will be described first followed by any differences to be found in the Dragon 32.

The CPU

The 6809 CPU (IC38) is, in my opinion one of the best microprocessors around at the moment. It is easy to program and has the potential to be very fast. I even prefer it to the current super microprocessor from Motorola, the 68000! In the Dragon the 6809 is used in a very standard way apart from being teamed up with the 74LS783 SAM chip (IC39) which provides all of the system control – but more of this in the next section. The system clocks E and Q are generated by the SAM and the standard frequency is .9MHz, which makes the standard Dragon a single speed device. However, it is possible to double the clock rate under software control.

It is interesting to notice that the address bus isn't buffered, because the SAM drives the RAM address lines and the other devices using the address lines impose very little load. The data bus is buffered by IC25 but notice that this only buffers the output of the RAMs, and so the data bus available at the expansion connector should be treated with care. A manual reset is generated by SW1 and a power-on reset pulse is generated by the simple strategy of charging a capacitor (C8) through a resistor (R17). The reset pulse is also distributed to the two PIAs (IC5 and IC8) and the ACIA (IC11). Of the remaining control lines only the two interrupts are used. FIRQ (Fast Interrupt ReQuest) and IRQ (Interrupt ReQuest) are connected to PIA IC8 and PIA IC5 respectively. The HALT line is made available on the expansion connector but is otherwise unused.

System Control – the SAM

In many ways the SAM (IC15) is the chip that makes the Dragon the Dragon. It not only generates all of the system clocks, but controls both the CPU's and the video generator's access to the RAM so that they are quite happy to share it. As well as these two important tasks it also looks after the decoding of the addresses and hence determines the Dragon's memory map and it automatically refreshes the dynamic RAM. All-in-all the SAM is an incredibly useful chip that deserves to find its way

into other designs. The tasks that the SAM carries out fall into three categories:

- addressing and refreshing the dynamic RAMs
- feeding data to the video controller
- address decoding

and as they are relatively independent, each one will be described in turn.

SAM and RAM

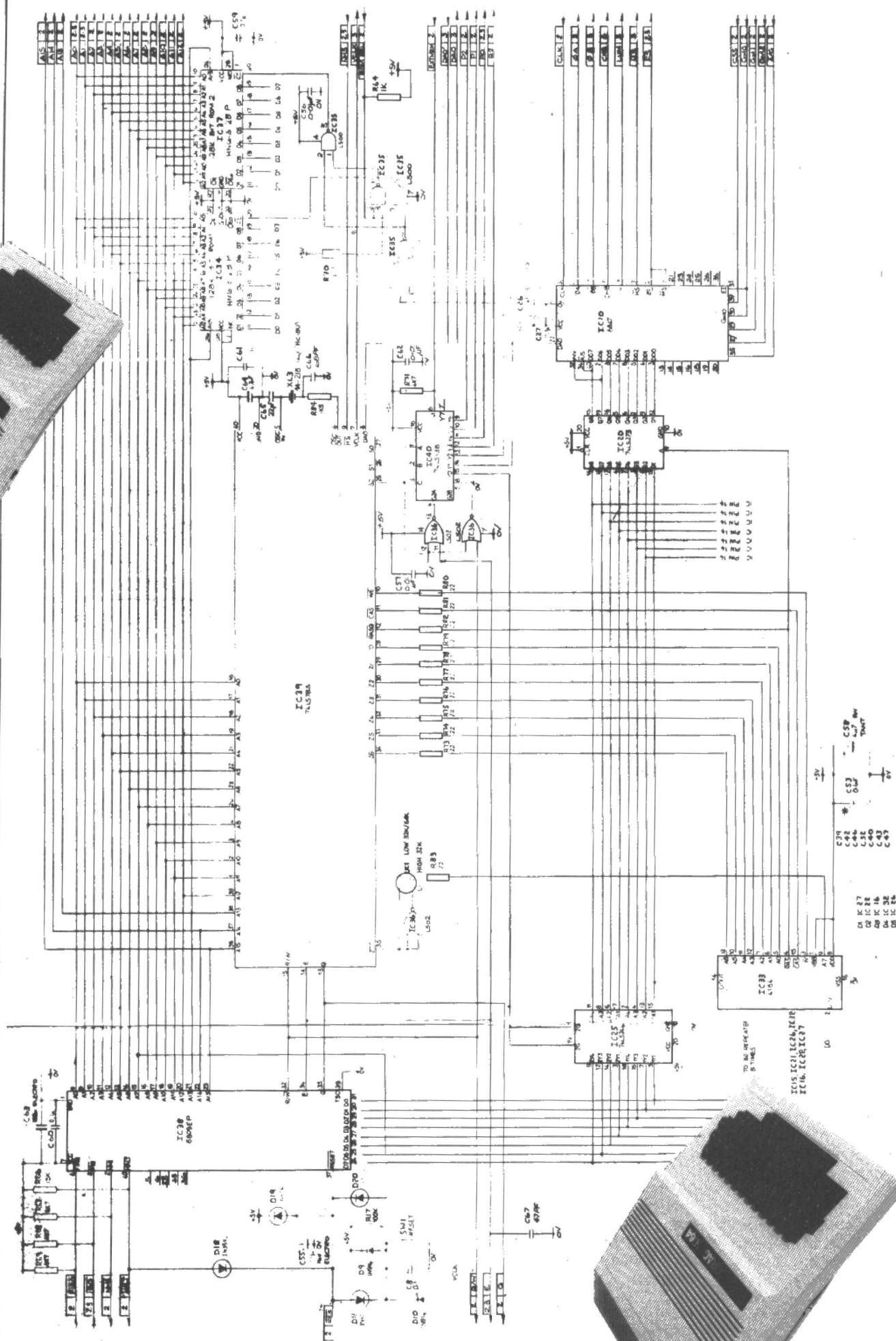
From the circuit diagram the most obvious thing that the SAM does is to multiplex the address lines to the RAM. A dynamic RAM has to be addressed in two stages because it is organised as a square array of storage locations. To select a given bit you first have to specify a row address and then a column address: For example a 16K dynamic RAM needs a seven bit row address followed by a seven bit column address. Normally the division of the 14 bit address (needed to address 16K) into a row and column address requires a number of different chips and some complex timing considerations, but in the case of the Dragon everything is handled by the SAM. The 16 address lines, A0 to A15, are connected to the SAM which converts them to eight row and column address lines, Z0 to Z7, and two timing signals, CAS (Column Address Strobe), and RAS0 (Row Address Strobe 0). The column and row address strobes indicate whether the lines Z0 to Z7 hold a column or row address.

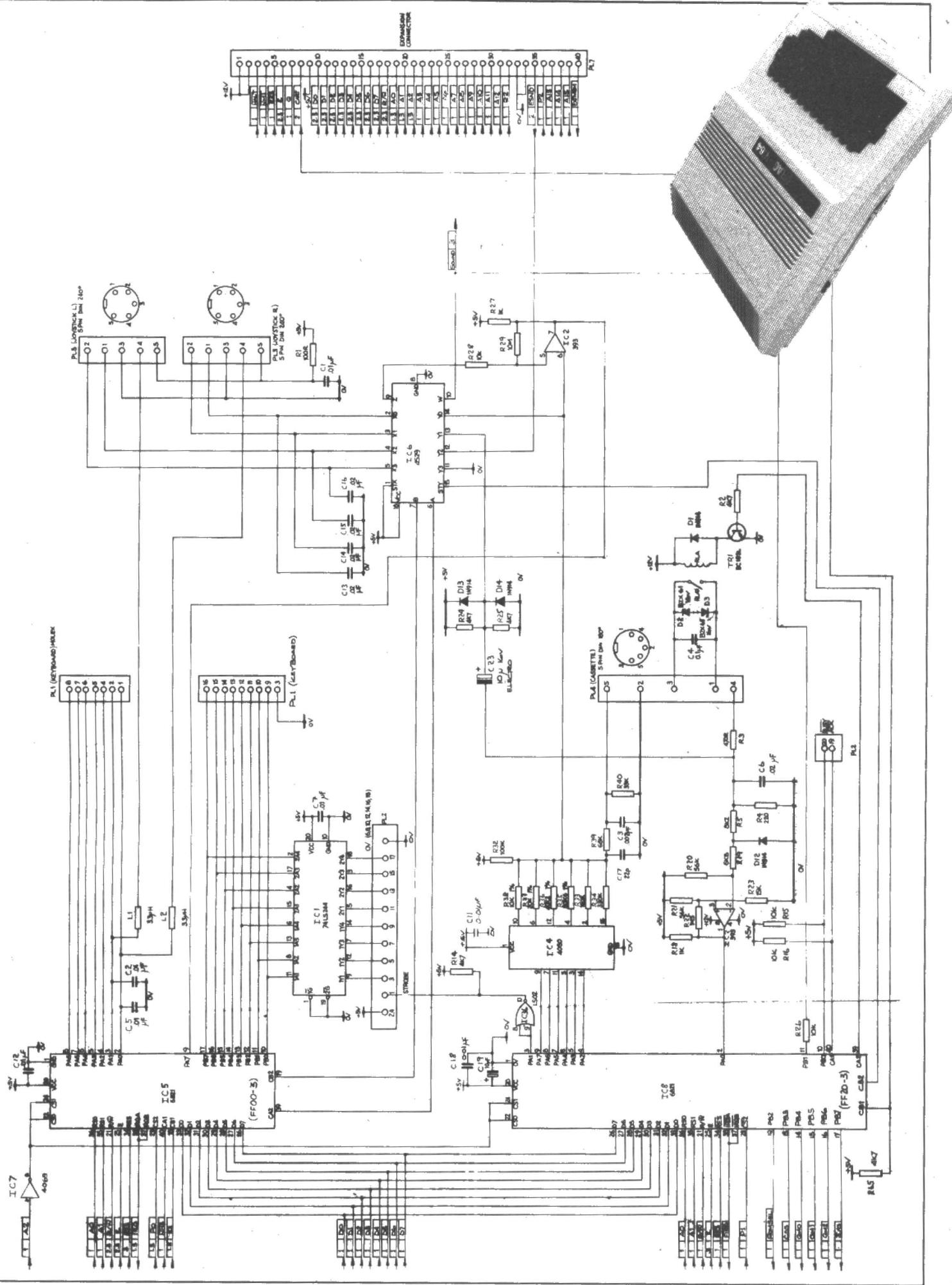
SAM and VIDEO

To understand how the Dragon's video system works you need to know something of how a conventional video system works. The problem in designing a memory-mapped video system is to find a way for the CPU and the video to share the same portion of RAM. The CPU needs to read and write the RAM so that it can run programs and change data, but the video controller also needs to read data from the RAM to generate a display. The video controller needs to access RAM at fixed and regular intervals and in many machines this is achieved by giving it a higher priority than the CPU – ie if the CPU wants to use the RAM while the video controller is reading data then the CPU is forced to wait. As the video controller spends a lot of time reading from RAM this can slow the system down considerably. In the Dragon the SAM chip manages all access to the RAM in such a way that the video controller can read data while the CPU is busy doing something that doesn't involve the RAM. In other words, the memory access is interleaved in such a way that there is no delay because the CPU is waiting for the RAM to be free. To achieve this synchronous operation the video generator is not allowed to address the RAM directly – it does have address lines but they are not connected to anything! The SAM chip 'knows' the order that the video controller needs to read data from the RAM, and so it generates the necessary row and column addresses and timing signals that isolate the data bus from the CPU (IC25 is a tri-state buffer) and

ELECTRONICS & COMPUTING

DRAGON 64 CIRCUIT DIAGRAM





latch the data (IC20) so that the video controller has time to use it. Obviously there has to be some way that the video controller and the SAM can be synchronised and this is achieved by the SAM generating and controlling the video clock. The video controllers address line DA0 is also read by the SAM to check that it is synchronised exactly.

Address decoding

The 16 address lines applied to the SAM are also used to select either the internal registers of the SAM or up to eight external devices. However before these eight device selects are brought out of the chip they are multiplexed onto three lines – effectively providing a three-bit address for the eight devices. This three bit address is decoded back to eight device select lines by a 74138 3-to-8 line multiplexer (IC40), the outputs of which are labelled Y0 to Y7. These lines are used to select between the RAM (Y0), two different ROMs (Y1 and Y2), the external cartridge ROM (Y3), two PIAs (Y4 and Y5) and a spare cartridge select (Y6). Device select Y7 is not used.

This completes the hardware description of the SAM chip but it is hardly a complete description of the SAM. To fully understand how it works you have to know something about the way it can be programmed to produce different memory maps. Different video display modes, handle different types of memory and generate a range of clock speeds. You can find out most of this detail from its data

TABLE 1

A/G – selects between alpha/semi-graphics and full graphics
A/S – selects between alpha and semi-graphics
E/1 – selects between external and internal character generator
INV – invert display
GM0 – Graphics mode bit 0
GM1 – Graphics mode bit 1
GM2 – Graphics mode bit 2

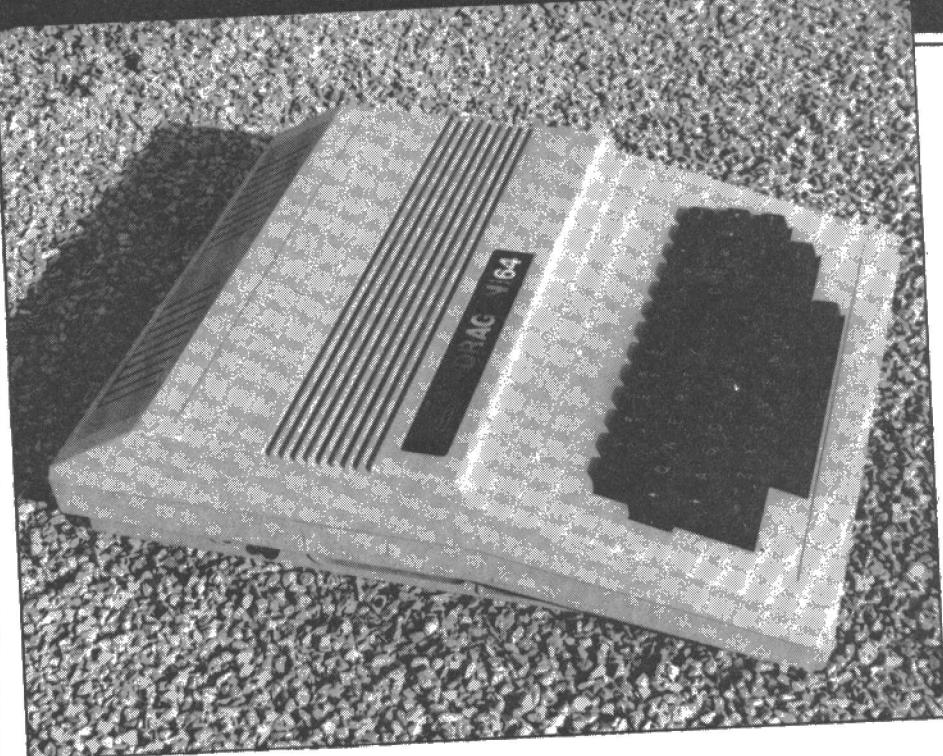
sheet or from the reference given at the end of this article.

ROM

The Dragon 64 has two 32K ROMs, 613128 (IC34 and IC37) holding Microsoft BASIC and the I/O routines. The 12 address lines are applied directly to the ROMs; the way in which the two chip selects, Y1 and Y2, are derived from the SAM has already been described. A circuit is included that allows the ROMs to be switched out of the memory map to give a full 64K of RAM. The ROM switching is controlled by output line PB2 on one of the PIAs (IC8).

The video controller

The 6847 (IC12) is a close relation of the 6845 used in the BBC Micro but its performance is very different. The 6845 was never intended to be used as a general purpose colour graphics generator but the 6847 was. For example, if you look at the



6845 data sheet you will see that there is no provision for colour at all, whereas the 6847 has a number of colour graphics modes. Unfortunately the 6847 isn't as good at text display as it is at colour graphics, being limited to a 32-character 16-line upper case only display. (It is possible to get 57 characters to a line using special software but this uses a 6847 high resolution mode.) As already described, the video controller's access to the RAM is controlled by the SAM chip. What is less obvious is that the graphics mode depends both on the way that the SAM chip is programmed and on the display mode that the video controller is in. The SAM chip must know the order and the number of bytes that the video controller needs in any display mode, but it can supply the same byte more than once so producing a lower resolution than the video controller's mode would suggest. This interaction between the SAM and video controller is too complicated to go into here but it is worth being aware of.

The video controller is programmed by setting the levels on a number of input lines, and unlike most other controllers, doesn't appear as a number of registers in the Dragon's memory map. However, some of the input lines are set by outputs from one of the PIAs (IC8) and so the display modes are still under software control, if indirectly. The control lines are shown in **Table 1**.

The A/G, GM0, GM1 and GM2 lines are all controlled by one of the PIAs (IC8) and so can be set under software control to produce any given graphics mode. The A/S line is connected to bit 7 of the data bus and so while in alpha/semi-graphics mode any character with an ASCII code greater than 127 (ie bit 7 set) will display as a semi-graphics block character. Similarly, INV is connected to bit 6 of the data bus, and so while in alpha/semi-graphics mode any character with an ASCII code with bit 6 set will display as a green character on a black

background. The final control line E/I has interested me since the Dragon was first produced – it selects between an external and internal character generator. With its help it should be possible to give the Dragon lower case characters but so far I've not come up with a design that doesn't involve extensive modifications. In the standard Dragon 64 this control line isn't used and is simply tied to the GM0 line.

The outputs of the video controller are shown in **Table 2**.

TABLE 2

LUM	six level analogue output containing composite sync and four levels of video luminance
OA	Three level colour signal
OB	Four level colour signal (Blue Yellow)
CUD	Chome bias, a DC reference
FS	Field sync
HS	Horizontal sync

The colour and luminance information are combined by a number of chips and three transistors to produce a standard composite PAL colour signal. If you would like to know a little more about this and how to improve the black and white Dragon display then see *E&CM* May 1985 page 47.

Reference: "Anatomy of the Dragon", Mike James, Sigman Technical Press, 1983. This provides a comprehensive guide to the Dragon's hardware and the way that it inter-relates with software. It is available from the *E&CM* Book Service.

Acknowledgement: Thanks are extended to Compusense who supplied the diagram used in the preparation of this article. Readers who have requirements for Dragon hardware or software are advised to contact Compusense (01 882 0681).

Next month – the PIAs and the Dragon's I/O.

Super Printer

How to print digitised picture posters using Epson FX and MX dot matrix printers. John Yau explains in the final article of the series.

The final set of programs in the software package enables the digitised picture posters to be printed using the Epson MX/FX range of dot matrix printers. For each picture element a rectangular cell of dots is printed, its optical intensity depending on the value of the data. 'Windows' into the picture can therefore be printed. The complete poster is constructed by assembling these picture segments.

The printer driver program shown in **Listing 1** prints a rectangular cell of 8 x 22 dots for each picture element. The latter cell size was chosen to preserve the picture's aspect ratio. In the 'giant poster' mode the program prints four identical cells to form a larger cell for each picture element.

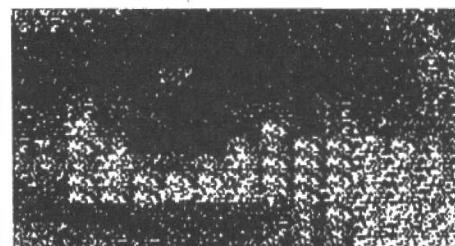
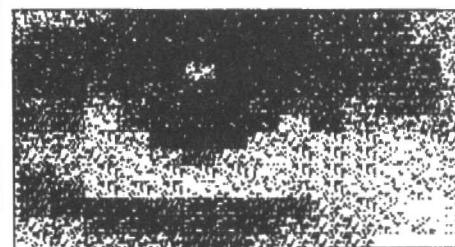
'Procinitprinter' sends control codes to the printer, enabling 8 dot line spacing and disabling the automatic form feed. All bit-image printing is done via the 'ESCL' dual density mode. If your printer does not have auto line feed then line 840 should be altered to VDU 1,13,1,10.

When a picture element is processed, it is first limited to a maximum of 127 and then presented as an index into the grey character look-up table CTAB%, **Listing 1**. The data retrieved from CTAB% indexes into the desired grey character cell to be printed as held in CH%. The data for CH% is generated by the character generator program as shown in **Listing 2**. As each advancing cell is created, the program turns increasingly more dots in the cell into black in a random manner. A 'card shuffling' algorithm is used to ensure that a dot is not splayed into a part of the cell that is already black. The resulting file is then saved onto disk to be loaded in by the

printer driver program when required. Note that the character set only needs to be generated once, unless one requires pictures of varying textures.

Perhaps the most important aspect of the printer driver program is the look-up table for the grey characters. The file CH% contains the character cells in a sequential form of linearly increasing greyness. By altering the parameter's brightness, resolution, threshold and step the mapping function from the picture elements to the grey characters can be varied, enabling a diverse range of effects to the printed picture. The effect of the latter parameters are illustrated in printouts (right).

Brightness simply moves the linear graph left or right and thus controls the



Illustrating the effect of the software and brightness controls.

'... although the software presented is extensive there is scope for experimentation with colour, error detection, etc.'

overall amount of black and white (if step=0). It was found that pictures printed with such a linear relationship tended to give diffuse 'photographic' reproductions as expected. In order to enhance the picture, parameters 'TH' for threshold and 'ST' for step were added so that once a picture element's grey level was above the threshold value it would be printed much darker than usual according to how much the linear graph has been 'stepped' up. The value of threshold tends to 'silhouette' the printed picture when low and tends to enhance the picture generally when higher. However the extent to which it effects the

picture depends on the value of step, see **Figure 2** (last month).

Experimentation with the look-up table parameters can be undertaken on a small section of the picture until the desired effect is obtained. Once the poster sections are printed they can be assembled. A good method is to tack the ink side of the picture 'sections' together with bits of masking tape. The sections are then turned over and permanently joined with sticky tape. Once that has been done, the masking tape can be removed.

LISTING 1. Printer driver.

```

10 REM FX/NX88 Printer Driver
20 MODE7
30 DIM CX 2928,VBUFFX 19200,CBX 18,NAMEX 7,CX 12,CTABX 128
40 PROCAssembly:PROCtitle:PROCloadchar:PROCloadfile:PROCinitprinter
50 CTZ=0:RESX=7:STZ=0:THX=127:PROCloadctab
60 ONERROR GOTO20
70 REPEAT
80 PROCtitle
90 PRINT "Contrast";CTZ
100 PRINT "Resolution";RESX;"bits"
110 PRINT "Threshold";"THX"
120 PRINT "Step";"STX"
130 PRINT "L" " load file."
140 PRINT "R" " change resolution."
150 PRINT "C" " change contrast."
160 PRINT "P" " print picture section."
170 PRINT "?"
180 A$=GETS
190 IF A$="L" PROCloadfile:GOTO240
200 IF A$="R" PROCcontrast:GOTO240
210 IF A$="C" PROCcontrast:GOTO240
220 IF A$="P" PROCdump:GOTO240
230 GOTO100
240 UNTIL FALSE
250 END
260 DEFPROCtitle
270 CLS:PROCOPD("Digitiser FX88 Printer Driver")
280 ENDPROC
290 DEFPROCinitprinter
300 VDU2,1,27,1,65,1,8,3
310 VDU2,1,27,1,79,3
320 ENDPROC
330 DEFPROCloadchar
340 PRINT "Load grey scale characters"
350 INPUT"Filename",F$
360 P=OPENIN F$
370 FORIX=0TO (22*128-1)
380 CHZ21Z$BGETRP
390 NEXT
400 CLOSE#P
410 ENDPROC
420 DEFPROCloadfile
430 PRINT "Input picture file."
440 INPUT"Filename",F$
450 !CBZ=NAMEX:#NAMEZ=F$
460 !CBZ+2=VBUFFX
470 !CBZ+18=19200
480 !CBZ+6=8
490 CALL1 loadpic
500 ENDPROC
510 DEFPROCdump
520 PRINT "Normal or Giant size. ?"
530 REPEAT:A$=GET:UNTIL(A$=70)OR(A$=71)
540 IF A$=71 SIZEZ=2 ELSE SIZEZ=1
550 PRINT "Now input window to be printed."
560 PRINT "Left X (0-95)";INPUTIX
570 XM=IX*40/SIZEZ-1;IF XM>95 XM=95
580 PRINT "Right X (max=95)";INPUTIX
590 PRINT "Top Y (0-199)";INPUTIY
600 PRINT "Bottom Y (max=99)";INPUTIY
610 PROCprint
620 ENDPROC
630 DEFPROCassembly

```

```

640 PX=CX
650 !OPF10
660 .loadpic
670 LDA #FF1LDX #CBX MOD 256:LDY #CBX DIV 256:JSR LFFD01RTS
680 J
690 ENDPROC
700 DEFPROCprint
710 DOTSX=(XZ-X1X+1)*SIZEZ*22
720 LX=DOTSX MOD 256:HZ=DOTSX DIV 256
730 VDU2
740 FORrowZ=Y1XTOY2X
750 FORCX=1TO81ZEZ
760 VDU1,27,1,76,1,LX,1,HZ
770 FORcolX=X1ZTOZ22
780 addX=colX*256+rowX
790 levelZ=YBUPF27addr2
800 IF(levelZ>127) levelZ=127
810 codeX=CTABX?levelZ
820 FORX=1TO81ZEZ:PRNCgenviduode(codeX):NEXT
830 NEXTCX
840 VDU1,13
850 NEXTHZ
860 NEXTrowZ
870 VDU3
880 ENDPROC
890 DEFPROCgenviduode(ax)
900 LZ=aZ*22
910 FORCX=1TO1Z+22
920 ENDPROC
930 DEFPROCcontrast
940 PROCtitle
950 PRINT "Brightness level (-100 to +100)";:INPUT CTX
960 IF(A$=CTZ)I$=0:GOTO950
970 PRINT "Threshold (0-127)";:INPUT THZ
980 IF(THZ<0)DRT(HZ)>127) THEN970
990 PRINT "Step (0-127)";:INPUT STX
1000 IF(STX<0)STX>127) THEN990
1010 PROCloadctab
1020 ENDPROC
1030 DEFPROCbitwise
1040 PRINT "Type resolution 1-7bits."
1050 INPUT "Value ",RESX
1060 PROCloadctab
1070 ENDPROC
1080 DEFPROCloadctab
1090 RSZ=17-RESZ
1100 IF CTX>0 THEN1100
1110 FORMZ=$TOCTX:CTABX?WZ=0:NEXT
1120 FORMZ=CTX TO127
1130 BX=MX-CTZ-(WZ>THZ)*STX
1140 IF BX>127 BX=127
1150 CTABX?WZ=(BX DIV RS)+RS
1160 NEXT
1170 GOTO1240
1180 FORMZ=$TO127+CTX
1190 BX=MX-CTZ-(WZ>THZ)*STX
1200 IF BX>127 BX=127
1210 CTABX?WZ=(BX DIV RS)+RS
1220 NEXT
1230 FORMZ=(128+CTX) TO127:CTABX?WZ=127:NEXT
1240 ENDPROC
1250 DEFPROCOPD(A$)FOR IZ=0 TO 1:VDU &9D84;&BD83:PRINTSPC(16-LENA DIV2)A$:NEXT
1260 ENDPROC

```

LISTING 2. Character generator.

```

10 REM GREY SCALE CHAR GENERATOR
20 REM Generates 128 shades in 22x8 cell
30 REM (C) John Yau December 1984
40 REM
50 MODE7
60 DIM XX 176,CH(22,8),BUFFX 3000
70 PROCinit
80 PROCaAssign
90 END
100 :
110 DEFPROCinit
120 FORIX=1TO176:XX?IX=IX:NEXT
130 PTZ=0
140 PROCOPD("Grey Scale Char. Gen.")
150 PRINT "Calculating..."
160 ENDPROC
170 :
180 DEFPROCswap(X1,X2)
190 TEMP=XX?X1:XX?X1=XX?X2:XX?X2=TEMP
200 ENDPROC
210 :
220 PROCaAssign
230 FORI=1TO176:STEP(175/127)
240 PRINTI
250 PROCClear
260 IF I=1 THEN320
270 FORJ=1 TO I
280 PROCsswap(J,INT(RND(176-J)+J))
290 :
300 YYZ=((XX?J)-1) DIV 22:XXZ=XX?J-22*YYZ
310 NEXT
320 PROCoutchar
330 :
340 PRINT:INPUT"Filename",F$
350 IFLEN(F$)>7 PRINT"Filename too long!":GOTO340
360 P=OPENOUT F$
370 FORIX=0TOPTZ-1:PUT#P,BUFFX?I%:NEXT
380 CLOSE#P
390 ENDPROC
400 :
410 DEFPROCoutchar
420 FORX=1TO22
430 BYTE=CH(X,1)+CH(X,2)*2+CH(X,3)*4+CH(X,4)*8+CH(X,5)*16+CH(X,6)*32+CH(X,7)*64+CH(X,8)*128
440 BUFFX?PTZ=BYTE:PTZ=PTZ+1
450 :
460 ENDPROC
470 :
480 DEFPROCclear
490 FORKX=1TO22:FORLX=1TO8
500 CH(KX,LX)=0
510 NEXT,
520 ENDPROC
530 DEFPROCOPD(A$)FOR IX=0 TO 1:VDU &9D84;&BD83:PRINTSPC(16-LENA DIV2)A$:NEXT
540 ENDPROC

```

Conclusion

The process described in this article allows pictures from a video source to be lifted and produced as digitised posters at very little extra cost. Although extensive software has been presented for the digitiser there is scope for further experimentation. For example, one might try to use a different graphics mode in the viewer program to include colour in the picture element cell. The printer driver program may be mod-

ified so that it performs edge detection, resulting in the printing of contours of equal optical density. Such a process can be applied to the printing of 'painting by numbers' templates!

Acknowledgements

I wish to thank the following people for their help with this project: David Beale of High Force Electronics, Alistair Houstin, Ian Anderson and Heriot-Watt University.

The author is able to supply the following

software for the Video Printer project:

Frame Grab, Mode Zero Viewer, Printer Driver and Character Generator for the Printer Driver.

The software will be supplied on 40 track S/D 5 1/4" disks. When ordering readers may choose an optional picture file. Please specify either the Marilyn Monroe, Robert Redford or Paul Newman picture files.

The software costs £12, fully inclusive and may be ordered from the author at 7 Maurice Place, Edinburgh, EH9 3EP.

A BASIC/MACHINE CODE INTERFACE

Joe Pritchard explains how to pass Spectrum string variables and constants into machine code, and gives a couple of applications.

One of the annoying features of the Sinclair Spectrum is the lack of a decent BASIC/Machine Code interface; the USR call allows us to execute our programs but gives us no straightforward method of passing parameters over to the machine code program. Although numeric variables can be poked into memory at addresses where they can be picked up by the routine, it's not so easy with strings.

In an article in the May 1984 edition of *E&CM*, Mike James showed how to pass numeric variables to machine code programs using the FN call. In this article I will describe a similar technique that enables you to pass string variables and constants over to machine code programs using the same call.

The technique requires a USR call to the address of your machine code program as the argument of a user defined function.

'... a technique using the FN call ...'

During the execution of a function, the system variable DEFADD, at address 23563, holds the address of a block of memory that contains data about the parameters passed over to the function. It is clear that if we make our machine code call part of the function, the parameter

information block will be available to us. For numeric parameters, this block holds the current value of each numeric parameter in the function call. Thus in:

```
10 DEF FN A(l)=USR 60000
20 PRINT FN A(w)
```

the sequence of bytes pointed to by DEFADD will hold the current value of the parameter passed with the function, in this case the value of 'w'. So we can write a short piece of machine code to pick up this value and make it available to the machine code routine.

With string parameters, the area of memory pointed to is arranged differently; it doesn't hold the contents of the string, but holds the address and length of the string being passed as a parameter, as shown in **Table 1**. This assumes that a string parameter is the first parameter in the FN call.

TABLE 1.

(DEFADD) single char. variable name	
"\$"	
?	
?	
low byte of string address	
high byte of string address	
low byte of string length	
high byte of string length	
")" or ","	

If the last entry is the code for a ")" then it indicates that this particular parameter

was the last one. If the code for "," is found here, it indicates a second parameter. The two bytes called "?" show my ignorance, because I don't know what these signify!

Armed with this knowledge, we can gain access to BASIC string variables or constants passed as parameters in the function call, and so we can write string handling utilities in machine code. For further details of the numeric parameter details, consult Mike James' article referred to above.

LISTING 2. BASIC test program.

```
10 DEF FN A(R$)=USR 60000
20 LET L=FN A("hello")
30 LET k$="hello"
40 LET L=FN A(k$)
50 PRINT k$
```

With regard to the string information block shown above, the first entry holds the name of the string variable that was passed when the function was called. The address entry of the table will point to the first character of the string in memory. If a string constant is passed to the routine, as in:

```
LET L=FN A("hello")
```

then the address entry of the table will point to the "h" of "hello" in the above

LISTING 1. Assembler conversion routine.

CP	123		INC	HL
JP	C,OK		INC	HL
LESS	RET	;if we get here then	INC	HL
		;the character wasn't a lower	LD	C,(HL) ;get low byte of address
		;case letter so back to BASIC	INC	HL
OK	RES	5,A ;do it	LD	B,(HL) ;get string add. into BC
	LD	(HL),A ;put modified character in the	PUSH	BC
	RET	;string and return to BASIC	POP	HL ;string add. now in HL
	ORG	60000	LD	A,(HL) ;get first character
DEFADD	EQU	23563	CP	97
	LD	HL,(DEFADD) ;get address of block	JP	C,LESS ;if code less than 97
INC	HL			;it's not a lower case letter

SOFTWARE

LISTING 3. Case change.

```

        ORG 60000
        DEFADD EQU 23563
        LD HL,(DEFADD) ; get the address of information
        INC HL
        INC HL
        INC HL
        INC HL
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL) ;address of the string in BC
        PUSH BC ;save it
        inc hl
        LD C,(HL) ;low byte of length of string
        INC HL
        LD B,(HL) ;now string length in BC register pair
        POP HL ;recover address of string
        LOOP LD A,(HL) ;get char from string
        CP 97
        JP C,LESS
        CP 123
        JP NC,LESS
        RES 5,A ;if we get here, the character is a
                  ;lower case letter that we must alter
                  ;so do it
        LD (HL),A ;put character into string
        INC HL ;bump up pointer to next character
        DEC BC ;decrement counter
        LD A,B
        OR C ;check the counter for zero
        JR NZ,LOOP ;if more characters, do it again
        RET ;otherwise, back to BASIC

```

LISTING 4. INSTR.

```

        LD HL,(DEFADD)
        INC HL
        INC HL
        INC HL
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL) ;get the address of x$
        LD (STARTX),BC ;save it
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL) ;get the length of x$
        PUSH HL ;temporarily save HL
        LD HL,(STARTX)
        OR A ;clear the C flag
        ADC HL,BC ;HL now points to last char of x$
        LD (LENX),HL ;save it
        POP HL ;now recover HL
        LD DE,6
        OR A
        ADC HL,DE ;HL now points to low byte of
                  ;the address of y$
        LD C,(HL)
        INC HL
        LD B,(HL)
        LD (STARTY),BC ;save address of y$
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL) ;y$ length now in BC
        OR A
        LD HL,(STARTY)
        ADC HL,BC ;HL now points to last char of
                  ;y$
        LD (LENY),HL ;so save it
        LD HL,(STARTX)
        LD DE,(STARTY) ;set up these register pairs
                  ;to act as pointers to the
                  ;two strings
        LOOP2 PUSH DE
        POP BC ;put current position within y$ into
                  ;BC for return to BASIC
        LOOP LD A,(DE) ;get char from y$
        CP (HL) ;is it same as corresponding one in x$
        JR Z,OK ;yes it is
        PUSH HL ;no, it's not
        LD HL,(LENY)
        OR A
        SBC HL,DE ;have we gone through all the
                  ;characters in y$?
        JP C,NOP ;yes, so we've not found a match.
                  ;exit via NOP
        POP HL ;otherwise, restore HL to hold
                  ;the start of x$
        INC DE ;next character in y$
        JR LOOP2 ;round again
        INC HL ;we get here if we've matched a char.
        PUSH HL ;so preserve HL to point to next char.
        PUSH DE ;preserve position in y$
        LD DE,(LENX) ;now see if we've matched characters to
        OR A ;end of x$. If we have, then x$ is within
        SBC HL,DE ;y$
        JP NC,FOUND ;yes, all x$ matched
        POP DE ;no, so restore the registers
        POP HL
        INC DE ;bump up y$ pointer.
        JR LOOP ;round again
        NOP POP HL ;not found, so clear stack
        LD BC,00 ;put 0 in BC for the return to BASIC...
        RET ;and do it.
        FOUND POP HL ;success!
        POP HL ;clear up stack
        PUSH BC ;holds position in y$ of x$ as the
                  ;actual address in Spectrum RAM, so
                  ;we must convert this to a character
                  ;reference within y$
        POP HL ;exchange BC and HL
        OR A
        LD DE,(STARTY)
        SBC HL,DE
        PUSH HL ;now contains position in y$-1
        POP BC ;into BC ready for return to BASIC
        INC BC ;adjust the value
        RET ;all done!
        ORG 60000
        STARTX DEFW 0000
        STARTY DEFW 0000 ;defining some data storage
        LENX DEFW 0000
        LENY DEFW 0000
        DEFADD EQU 23563

```

statement; it will thus point into the BASIC program, and so if your machine code routine modifies the string in any way it will also modify the listing! If the parameter is a string variable, then the address entry will point to the string in the variables area of the computer memory. So, let's see how we can make use of this information, albeit in a trivial fashion. **Listing 1** shows an assembler routine to convert the first character of a string from lower case to upper case.

Assemble the above code to address 60000, and use the BASIC program (**Listing 2**) to test it out.

This will print "Hello" to the screen, and will modify line 20 of the program to

20 LET L=FN A("Hello")

What about more serious applications? Well, the ability to perform string manipulations in machine code is very useful in text manipulating programs, such as databases, adventure games and educational software. Listed below are two utilities that make use of the techniques demonstrated above.

Case Change (**Listing 3**) is an extension of the above to convert every lower case letter in a string to its corresponding upper case letter. This is useful when matching strings, such as commands in adventure games, where the case of the text doesn't really matter.

The routine accepts a single string parameter, and it can be a string constant or variable, the effects being as below:

LET L=FN A("hello") as a program line will become LET L=FN A("HELLO")

and a string variable will be modified in a similar fashion. Punctuation, etc. will not be modified by the routine.

The final program, INSTR (**Listing 4**) is a little long but is very useful. Again, the machine code should be loaded to address 60000. The routine accepts two string parameters:

```
10 DEF FN I(x$,y$)=USR 60008:  
  REM first 8 bytes are data  
20 LET c$="f":LET d$="abcdefg"  
30 PRINT FN I(c$,d$)
```

The value returned will be 0 if the string passed to x\$ is not present in the string passed to y\$. In the event of x\$ being in y\$, the value returned will be the position of the first character of x\$ within y\$. If x\$ turns up more than once in y\$, then the first occurrence will be the one that is returned to BASIC. So, in the above example, line 30 will return the value 6.

One application is obvious; that is, searching a data base. Indeed, the func-

tion was originally written for this purpose. A second application is in vetting input strings from the keyboard. A classic use is to test for Y/N response. If i\$ holds the character typed in at the keyboard, then

200 LET L=FN I(i\$,"YyNn")

will return 0 if the user types in any other letter than Y or N. Case is irrelevant. If the response was "Y" then the function will return 1 and so on.

So, on with the listing.

Once the code is in memory, you can see the routine in action using a program like that shown in **Listing 5**.

LISTING 5.

```
10 DEF FN I(x$,y$)=USR 60008  
20 INPUT A$:REM string to be searched for  
30 INPUT B$:REM string to be searched  
40 PRINT FN I(A$,B$)  
50 GO TO 20
```

Again, string constants or variables can be used.

I hope that this article has given you some "stringy" food for thought; machine code operations on character strings can be very fast indeed. The INSTR program above was virtually unuseable in BASIC, but has proved to be highly acceptable in data base searching and data validation routines now it's in machine code.

A Hardware Guide For The BBC Microcomputer.

A 260 A4 PAGE BOOK WHICH INCLUDES:-

- ★ Comprehensive circuit description.
- ★ Full step-by-step details on upgrading Model 'A' to Model 'B'.
- ★ Full details on fitting disc and speech expansions.
- ★ Full explanation of all link settings and functions.
- ★ Full of circuits, hints, tips and modifications (many previously unpublished).
- ★ Full manufacturers data sheets on all major IC's including:-
R6522, 8271, 6502, MC6845, MC6850,
SN6489AN, UPD7002, SAA5050, HM4816,
2764, 27128, 7400 series pin-outs.

ALL DEALER ENQUIRIES WELCOME

A MUST FOR BOTH ENTHUSIAST AND ADVANCED USER ALIKE

To order send cheque/PO for £11.95 plus £1.50 p&p (UK only)

(Overseas readers — printed paper rate surface mail — £3.00 p&p — air mail extra).



ORDER FORM

Please send copies of "A Hardware Guide for The BBC Microcomputer"

At £11.95 each plus £1.50 p&p (UK). I enclose cheque/postal order for £.....

NAME

ADDRESS

ECM06

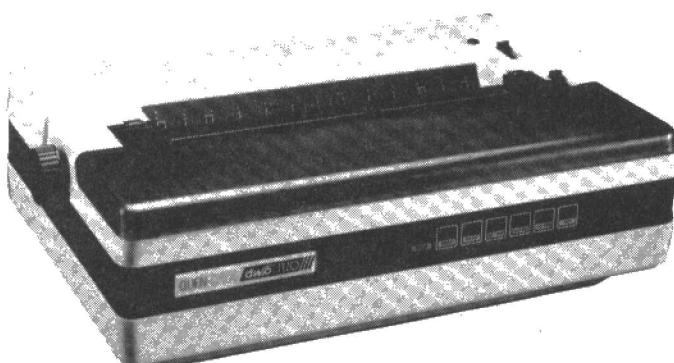
All orders and enquiries to

WISE-OWL,

Hull Innovation Centre, Guildhall Road, Queens Gardens, Hull HU1 1HJ.

THE RIGHT TYPE

Brian Alderwick and Peter Simpson review Quen Data's bargain basement daisywheel.



In microcomputing pre-history (some three years ago) the choice of a printer was simple: for 90% of users it had to be a dot matrix type (probably an Epson MX80 or Centronics). Other printers were either totally unsuited to home use or very expensive (daisywheels were priced at between £800 and £1600). The choice was thus easy and dictated by cost.

In the last year or so printer prices have fallen considerably. A bewildering variety of thermal, graphic, ink jet, dot matrix, daisywheel and laser printers offer speeds of between 10 cps (characters per second) and 2000 cps. Dot matrix printers are still the hobbyist's favourite because of their combination of speed, versatility and price – but the quality of print ranges from poor to average.

Some dot matrix printers are now supplied with an NLQ (Near Letter Quality) mode which, by using quadruple bit density printing, produces well formed characters, but this reduces printing speed to about one fifth of its maximum value and the print quality is still not comparable with a daisywheel printer. If the quality of printing is more important than the speed, a daisywheel now offers a real alternative. The Quen-Data DW1120 is the first of the

daisywheel printers to come down to a really affordable price: it retails at around £250, which is cheaper than many popular dot matrix printers.

Our review machine came in a strong cardboard box, with a mains lead (complete with 13 amp plug – a nice touch) daisywheel, ribbon cassette, dust cover, and multi-lingual instruction book. We connected it to a BBC micro via the parallel interface (the socket is a standard Amphenol type, the same as that used by Epson). A serial RS232C interface is available at extra cost, fitted with a standard 25-pin D socket. Like many other printers, the serial interface CTS line from the micro should be fitted to the printer DTR pin 20. If software handshaking is required then both ETX/ACK and X-ON/OFF protocols are provided.

First impressions are of a neat, good looking unit, but it is bulky compared with dot matrix printers. The size is governed by the platen which is a generous 13 inches long, which will enable large tables (spreadsheets?) to be printed.

The printer is 55cm wide x 34cm deep x 16cm high and it tips the scales at 9.5kgs.

The case is sturdy and shows no sign of cost cutting, and inside the case it was nice to see attention to detail such as the screening of ribbon cables.

Fitting the daisywheel and ribbon cassette is easy and a 'clean hands operation' unlike that required by some other daisywheels. After connecting the cables, the test mode can be used to check that

“... first impressions were of a neat and good looking unit . . .”

everything is working: pressing TEST on the front panel prints all 96 characters available on the daisywheel (if all is well).

Compatibility

The standard daisywheel supplied is Courier 10 pitch. Most suppliers of daisywheel printers (and typewriters) provide a 10 pitch wheel, ie 10 characters to the inch, which is odd because most wordprocessing is done in 12 pitch. The standard wheel has no £ sign, but Quen Data says this will be rectified in future.

With any new printer the availability of spare wheels and ribbon cassettes is an important issue. Fortunately, both of these expendable items are totally compatible with those available for the established QUME Sprint 3 printer. This compatibility is extended to the electrical interface and printer control characters that are all set to the QUME protocol. Thus printer drivers for word processors intended for QUME printers should function correctly with this printer. We fitted the QUME Prestige Elite 12 pitch daisywheel.

Software selection

Amongst the facilities available by software control are automatic underline, bold, shadow printing, half line up or down, setting the horizontal motion index, setting the vertical motion index, various tabs and setting the left hand margin position. These facilities can be utilised in a word processor printer driver to provide such enhancements as subscripts, superscripts and microspacing. When a word processor uses right justification, the line is usually padded out with spaces to fill the line length when the words do not fit exactly. Whole spaces are added between words as needed. As this printer can move the daisywheel 1/120" at a time with the horizontal motion index control, it is possible, with suitable software, to divide the additional spaces needed to achieve right justification equally between the words.

Operation

Paper loading is completely manual. A semi-automatic feed like that on the Juki 6100 would have been more convenient, and the paper would have stopped in exactly the same place each time freeing the user from this task. With a printer in this price range, however, not all facilities can be provided. Quen Data do offer a sheet feeder and tractor feed as options.

The front panel has six flush switches and four indicator lights. These are from left to right: first, a mains on indicator; secondly a switch marked ERROR complete with an indicator which lights paper end (when using continuous tractor fed paper), ribbon end, and cover open; third is the ON LINE indicator and switch which toggles on/off and either allows printing or not; the next three switches only work when in off line mode, and include SET PAGE which sets the top of form, PAGE ADV which advances the paper to the next top of form and LINE FEED which advances the paper one line at a time, but if depressed for more than one second runs continuously; the last switch and indicator provides the TEST option described above.

There are two banks of dual in line switches under the hinged cover and these are easily accessible. They set the default power up conditions for several options. The manual refers to switches A and B, but in fact they are marked on the board as 2 and 1 respectively. Switch B (or 1) is for selecting options on the RS232 interface (Baud rates etc.) but need not be changed.

for the parallel interface. The other 8-pole switch selects options which are common to both serial and parallel interfaces. These are Serial/Line print mode, Auto line feed (ie carriage return with or without line feed), Line feed selection (1/6" or 1/3"), Page length selection (11" or 12"), Pitch selection (10, 12 or 15 cpi) and Hammer impact selection (one of four, which allows the original and up to four copies to be printed at once).

"... an ideal printer for the home and small business user ..."

The functions of these switches are self explanatory except Serial/Line print mode, which should not be confused with the serial/parallel interface. In the serial print mode each character is printed as it arrives at the printer, thus making the computer/printer act as a typewriter (the in-built 256 byte buffer fills as the characters arrive if they are faster than can be printed). The printing action is uni-directional. In the line print mode the printer waits for a carriage return, line feed or form feed before printing starts, allowing it to work in a bi-directional, logic seeking manner. Therefore, for word processing, line print mode would be the most appropriate. It should be noted that if line print mode is used, some of the printer extended control codes are not

available, although the only two of consequence are horizontal motion index (ie microspacing is not possible) and absolute horizontal tab.

Speed and print quality

Printing speed is quoted at 18 cps (repeat) or 16 cps (for Shannon text). This slow speed is a consequence of designing a printer to sell at such a low price. Unfortunately, the cost of daisywheel printers seems to increase exponentially with printing speed. However, this speed is as good as a fast typist and if your text is not too long or repeated too often then it can be tolerated. The speed is comparable to that of similarly priced printers.

Daisywheel printers are traditionally noisy beasts, which require acoustic covers if operated within earshot. The Quen Data noise level is quoted at 60dBA which, although not quiet, is no worse subjectively than dot matrix printers.

Print quality is really the factor by which daisywheel printers are judged and there is no doubt that even NLQ dot matrix print quality comes nowhere near that of this machine. Incidentally, a QUME replacement carbon ribbon we used gave better quality print than the original supplied with the machine, and this is worth bearing in mind if seeking a demonstration.

Conclusion

The overall impression is of a sturdy, good

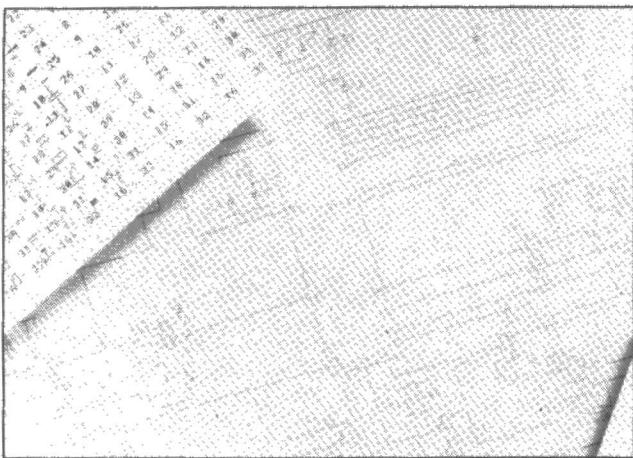
looking machine with acceptable printing speed and very good print quality. Direct comparisons at this level of the market are difficult because this printer has very few competitors. Also, long term reliability on a new printer is obviously difficult to assess. However, Quen Data says that dealers will replace the machine with a new one at any time up to 12 months from the purchase date if a claim is made under warranty. This is a welcome and almost unique service for a product of this price. The typical selling price of this printer is around £250 including VAT which is exceptional and cheaper than many popular dot matrix printers. For the home and small business user we would certainly give this a highly recommended label.

Data Sheet

Print type	daisywheel
Speed	18 cps
Interface	Centronics (RS232C option)
Weight	9.5Kg
Size	550 x 340 x 160mm
Platen width	330mm
Pitch	10, 12 or 15
No. characters	96
Noise level	60dBA
Price	£250 (inc. VAT)
Supplier	Quen Data Business Machines UK, 25 Clarke Road, Mount Farm, Milton Keynes MK1 1LQ 0908 649412

CIRKWIK SCHEMATIC DRAWING ON THE BBC MICRO

A CAD package orientated to the production of schematic drawings, such as circuit diagrams, flow charts, pipework diagrams, fluid logic diagrams and many similar professional and engineering applications.



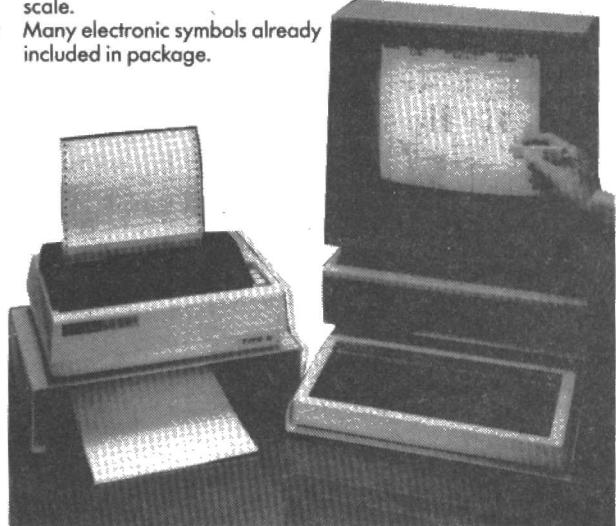
Datapen

CIRKWIK Program Lightpen
Only £19.95
CIRKWIK Program Tracker Ball/Lightpen £24.95

DATAPEN Lightpen £25 **MARCONI Tracker Ball £59.50**
S.A.E. for details of lightpen, CIRKWIK and other programs.

DATAPEN MICROTECHNOLOGY LTD. Dept. EC6, Kingsclere Road, Overton, Hants RG25 3JB Telephone: (0256) 770488

- ★ Lightpen or tracker ball driven
- ★ Virtual screen 8 x the BBC's mode 4.
- ★ Uses standard dot matrix printer in dual-density graphics mode to produce excellent quality diagrams.
- ★ Automatic parts list generation.
- ★ Up to 640 different symbols may be in use in any one diagram.
- ★ Total symbol library unlimited in size.
- ★ Create your own symbols with the lightpen on a highly magnified scale.
- ★ Many electronic symbols already included in package.



NEW SERIES THE FIRST SPECTRUM WORDPROCESSOR IN FIRMWARE PART THREE

The wordprocessor circuit consists of only four ICs, one of which – the ADC chip – is optional, and it is possible to run the whole board from the 5 volt supply line of an unexpanded 48K Spectrum. IC4, the ADC, is a CMOS device and draws a miserly 10mW of power. If you're buying fresh components, IC1 can be one of the new 74HC high-speed CMOS chips, the 74HCT138, which consumes far less current than the older 74LS138. IC2, the EPROM, will spend much of its time deselected, so it falls to IC3, the Z80APIO, to take the lion's share of the power consumption. But despite the low power consumption a dedicated 5 volt supply is recommended as this will help the computer run a little cooler and so prolong its life. There is also the ADC port to consider: using the +5 and 0 volt lines in analogue applications may inadvertently strain the power supply and if the computer is running on the same +5 supply, then a software crash would result.

Logic states

There is nothing particularly revolutionary in the way these four chips are connected together. IC1 decodes the top three address lines and so divides the Z80 memory map into 8 portions of 8K. Q1 is the second output of the decoder, and its signal goes low whenever an address in the range 2000H-3FFFH is selected. The low enables the shadow EPROM and is inverted by TR1 to provide the high signal necessary to deselect the Spectrum ROM. However, IC1 will not respond at all until it is given the go-ahead in the form of a high signal on pin 6. Normally pin 6 is held low by R7. Strictly speaking R7 shouldn't be needed if you're using a Z80APIO to control IC1: the PA2 line is LOW in its natural state immediately after switch-on and will stay that way until told to go HIGH by software. PA2 and PA1 are set at output lines, as are all of the PB lines, while PA0,3-7 are set up as inputs. The PIO is enabled by a LOW on A5. Its four internal registers are selected by values on A8 and A9, so creating the four addresses given in **Table 1**.

IC4 is enabled by the signal created by ANDing A6 with IORQ in the double diode configuration D2, D3. A6 LOW gives an address of 0111111 which is FFBF or 65471. In this application Vref (pin 8) and AGND (analogue ground, pin 7) are simply tied to Vcc and GND respectively, but you may wish to provide a stable reference

A CONSTRUCTION JOB

Richard Sargent has already explained how his Spectrum wordprocessor works and what it does. Now he gets down to building it.

TABLE 1. Internal register addresses

9 8 7 6 5 4 3 2 1 0	FCDF 65735 PADATA
0 0 1 1 0 1 1 1 1 1	FDDF 64991 PACONTROL
0 1 1 1 0 1 1 1 1 1	FEDF 65247 PBDATA
1 0 1 1 0 1 1 1 1 1	FFDF 65503 PBCONTROL

voltage to pin 8 and provision for this has

The "conversion done" signal, INSTR, is been made on the printed circuit board. | fed to PA7 of the PIO from which it can be

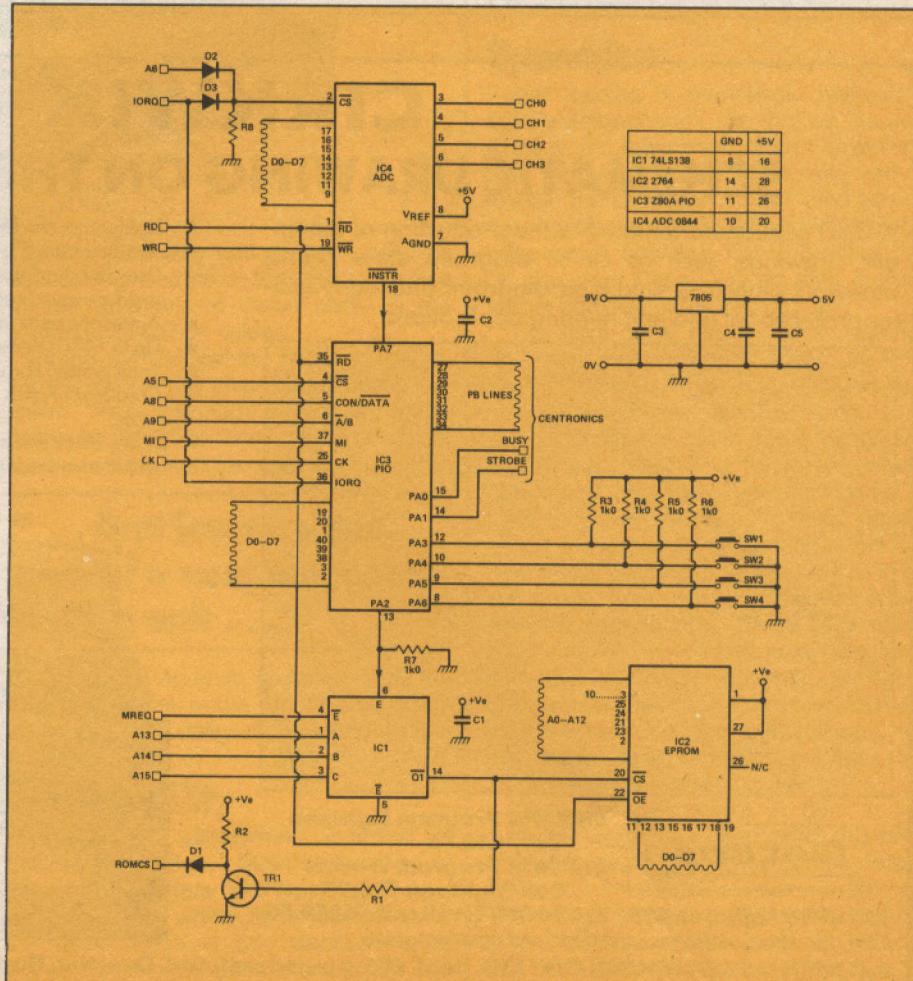


Figure 1. The WP's circuit diagram.

sampled by software if need be.

Three in one

The circuit can be built on veroboard if the wiring-pen method of construction is used. There are rather too many address and data line links to make the hook-up wire method a reliable option. However, a PCB pattern is available, and it may be used in one of two ways. It's really three patterns on the one board. As three separate boards, each fits into the casing of one of the cheap trackballs currently on the market for the BBC micro. This is a fiddly task, but the neat end result makes for an uncluttered desk! Alternatively, the board can be left uncut and placed in a conventional box. This leaves you with the mechanics of connecting up the joysticks or trackball units in whatever way you want. Whichever style you opt for, electrical connections must be made between the 3 separate patterns.

Board A is the PCB which receives signals from the Spectrum and which also contains IC1 and IC2. Board B carries IC4 and the 7805 regulator. Board C holds IC3 and feeds the output to the printer. Data lines, selected address and selected control lines link the three sections together, while a few other wires connect to the joystick/trackball potentiometers and to a couple of switches. The circuit involving the two potentiometers is quite straightforward and is shown as **Figure 1**. Further information on linking the three boards together will be given next month. The foil patterns will also be published next month.

8K of software

Filling 8K of memory space with meaningful code may seem a daunting task, and so it would be if the entire 8K happened to be program instructions. In fact, a significant proportion of the space is filled with look-up tables. The wordprocessor itself con-

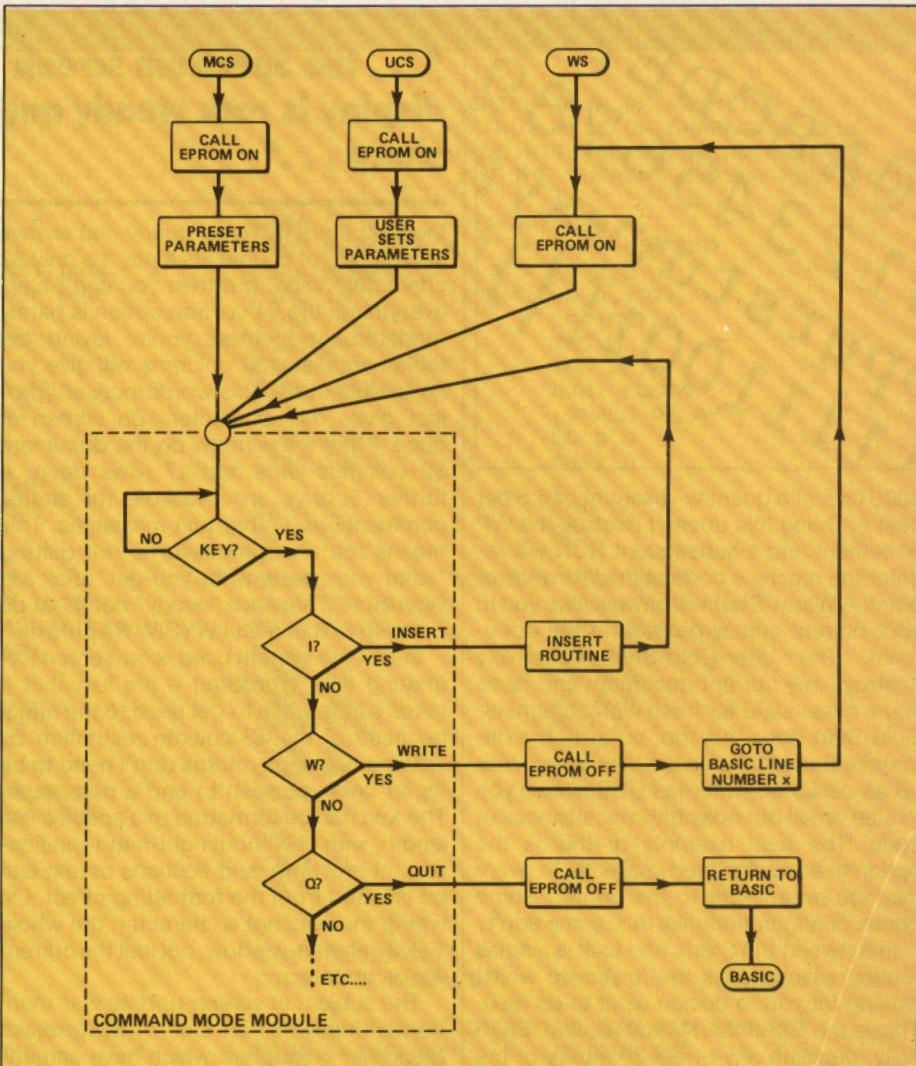


Figure 2. Flow diagram indicating the structure of the WP firmware.

available for this project, so there'll be no need for keyboard entry of thousands of hex numbers. There is code in RAM, but it's only a short BASIC handler dealing

The overall design

The basic wordprocessor occupies about 3K bytes; in this amount of space the following facilities can be adequately supported: fast vertical scrolling; insert mode typing; adjustable line length; justification; block copy; character deletion; line deletion; find and replace; whole or part-text printing.

I'll leave you to figure out what's missing – and don't assume that it will crop up somewhere in the other 5K either! This wordprocessor is not designed to be in competition with Brand X products: its strength lies in its specialities, which are: the ADC interface; hexadecimal to decimal conversions in situ; sorting, by specified field, or variable length records; manipulation of Epson RX80 graphics; Magnified text on any dot-addressable printer.

The structure of the wordprocessor program can be seen in the flow diagram, **Figure 2**. There are three entry points, Machine-Cold-Start, User-Cold-Start and Warm-Start, and one exit point, Quit. Taking these points in reverse order, Quit returns the user to Spectrum BASIC, from which direct commands such as `PRINT 399*8000*0.935` can be made (this supplies answers to burning questions of the moment such as what is the price of

... the word processor is not quite a what-you-see-is-what-you-get affair à la Wordstar . . . but it comes close . . .

sists of a number of modules which, in themselves, are of modest proportions. **Table 2** shows, very approximately, how the 8K EPROM is filled, and how much RAM is absorbed by workspace.

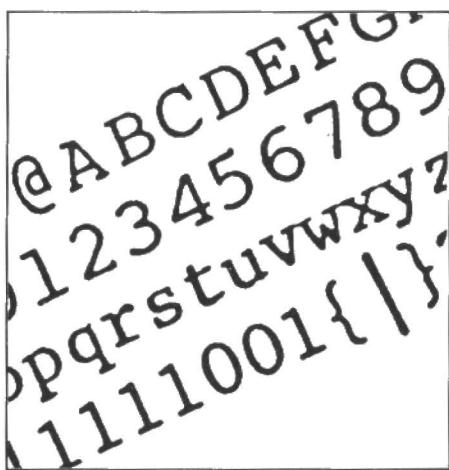
TABLE 2. EPROM contents

2000-2BFF	basic W.P.
2000-2FFF	number base conversion
3000-33FF	sorting routines
3400-3BFF	printer driver
3C00-3FFF	enlarged character-set
8000-EFFF	26K+ document (approx 4000 words)
EA00-EFFF	optional printer character-set
F000-FFFF	general workspace

Before anyone starts panicking, it's intended that pre-blown EPROMs will be

mainly with LOAD, SAVE and VERIFY. Other optional machine code patches can also be loaded in RAM to facilitate printer-driving modifications and experimental work, but such extras are entirely at the discretion of the user.

The length of code precludes any possibility of listing the contents either in source code or as a hexadecimal dump, and, by the same token, it won't be possible to explain every little nook and cranny of the wordprocessor. Instead, small sections will be highlighted to show the principles of operation. Together these will form a small (and I do mean small) wordprocessor which can fit into about 1K of RAM. Using this together with the PIO (IC3) provides a low-cost introduction to a Z80 wordprocessor – interesting from a machine code tutorial viewpoint but not suitable for writing novels!



8000 QLs at a quantity-discount of 6.5 per cent?). Doing this doesn't destroy the WP document and a subsequent RUN will re-enter the machine code at the Warm-Start point. A Warm-Start will always take you to your current cursor-position in the document. The User-Cold-Start effectively destroys any current document and gives you a clean slate for fresh work. Machine-Cold-Start does this too, but with the difference that MCS sets the Wordprocessor up as designed, whereas UCS allows you to alter small but possibly critical pieces of code. The best example of this is the option to add a line feed character to every carriage return sent to the printer. Some printer configurations like this, some don't.

In between the Starts and Quit is a small routine which scans the keyboard whilst waiting for you to decide what to do next. This is the Command Mode routine. Basically, pressing any of the 26 letters of the alphabet will cause the program flow to branch to a subroutine, only three of which are shown in the flow chart. Quit has been explained, Insert is the one used to enable

... the Spectrum screen display is rock steady and sharp ...

the best there is. Rock steady and always sharp. Yes, of course it's a pity it's only 32 characters wide, but then you can't have everything: the 64 column option is barely readable. And unless you're laying out tables and lists, 32 columns will give you no trouble at all. This wordprocessor project gives 544 characters on screen at once – that's a window on the document 32 columns wide and 17 rows deep. A further 5 rows are available for status, comments and other text messages. This means the wordprocessor is not quite a "what-you-see-is-what-you-get" affair à la WordStar. The word "nearly" needs to be prefixed to that awful WYSIWYG phrase.

In fact the system used is on-screen formatting (like WordStar) – it's just that it takes a little while to get used to the effect because of the 32 column restriction on viewing. Carriage returns don't need to be typed until you wish to end a paragraph. The text can be formatted to a specific line length, with or without right-hand justification, at any time, and sections of text can be isolated from the formatting effect. On typing (or deleting) a character, the whole 17 x 32 screen-window, not just the current line, is updated.

The subject of screen-update was mentioned briefly in Part One of this project. Basically, computers which have screens composed of individual pixels take some time to refresh with new information. The Spectrum has 192 x 256 pixels which is approximately 6K bytes of RAM to be

The keyboard

In contrast to the screen printer, the Spectrum key-reading routine is most efficient and doesn't need speeding up. The keyboard subroutine is at 02BFH in the ROM, and it is called 50 times a second by the Z80 Mode 1 interrupt system. So long as the interrupt is enabled, a machine code routine can grab the last keypress by the instruction LDA,(23560). The Spectrum wordprocessor merely makes use of this facility and adds a further routine to read the auxiliary switches that the hardware provides.

Listing 1 shows a test routine which, when run by RANDOMIZE USR 50000 prints keypresses onto line screen. It is a very rudimentary program – the cursor arrows and delete won't work, for example – but it serves to illustrate a number of points and is worth typing in if you have a hexloader handy. The program flow centres around TLOOP and there are three elements present in the loop: the code for the flashing cursor blob (28 bytes at TLOOP); the code for reading the keyboard (3 bytes at CONT2) and the code for printing to the screen (14 bytes in OUTVDU, 17 bytes in CONVERT and 13 bytes around CHL).

Register BC holds the ROW and COLUMN co-ordinates and so must not be corrupted by other routines. There are a number of areas where experimentation is worthwhile, provided, of course, that you can use an assembler to rearrange the code. The particular assembler used to create the source code in **Listing 1** doesn't use line numbers, so the code is referenced using its address.

C362: As the 16-bit counter held in COUNT cycles round the bits change state; the most-significant ones change slowly while the least-significant ones change extremely rapidly. Bit 7 has been chosen to toggle the cursor on and off and BIT 7,(HL) performs the necessary test to see whether 0 or 1 (cursor OFF and ON) is present. It follows that testing BIT 6,(HL) will double the rate of cursor flash, BIT 5,(HL) will quadruple it. To slow down the cursor flash, BIT 8 must be tested. To do this change LD HL,COUNT to LD HL,CHIGH and then use the test BIT 0,(HL).

... just because the word processor is in ROM doesn't mean that you can't tinker with it ... all routines are vectored through RAM ...

the text of the document to be created, and Write is one of two routines which sneaks back into Basic without the user being aware that this has happened. The Command Mode routine can cater for more than 26 commands, but the subroutines attached to them will need to be written and placed somewhere in RAM.

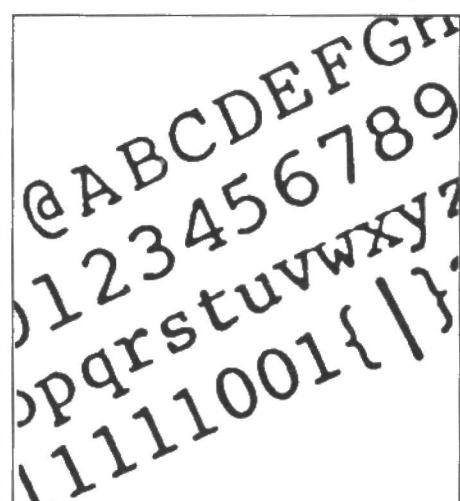
Don't be misled into believing that because the wordprocessor is in ROM, you can't tinker with it. All routines are vectored through RAM so you can intercept them and route the program flow to a replacement routine. For example, the screen display is 32 characters wide, but if you choose to you can trap the VDU-Driver vector and write your own 64-character screen routine.

The screen

The Spectrum screen display is amongst

loaded with information every few tenths of a second. Machines like the Acorn BBC, Memotech and Einstein win on this point because they have video-processors which can treat their screens as arrays of character blocks, usually 24 x 40, and that is less than 1K of RAM. They can write text to their screens virtually instantaneously, and change to pixel mode for graphic work. The Spectrum wordprocessor refreshes 4K bytes on every screen update.

It isn't possible to use the Spectrum's own print-to-screen routine at 0010H in the ROM because it is far too slow, dealing, as it must, with OVER and INVERSE conditions, STREAMS and ATTRIBUTES and uncle Tom Cobley. Machine code must attack the Spectrum Video RAM directly, and an example of how to do this is shown in **Listing 1**, which will be considered shortly.



LISTING 1. Test routine.

```

        ORG 50000
        LOAD 50000

; TEST
C350 3E01      LD A,1
C352 320A5C    LD (23562),A
C355 010000    LD BC,0
; MAIN LOOP -- FIRST DEAL
; WITH CURSOR SYMBOL AND
; FLASH RATE
C358 2AE1C3    TLOOP LD HL,(COUNT)
C35B 23        INC HL
C35C 22E1C3    LD (COUNT),HL
C35F 21E1C3    LD HL,COUNT
C362 CB7E      BIT 7,(HL)
C364 2808      JR Z,CONT
C366 11D1C3    LD DE,CURSHP
C369 CDA6C3    CALL OUTVDU
C36C 1806      JR CONT2
C36E 11D9C3    CONT LD DE,NOCUR
C371 CDA6C3    CALL OUTVDU
; NOW GET KEYPRESS INTO A
; KEY "0" IS THE ABORT KEY
C374 3A085C    CONT2 LD A,(23560)
C377 FE30      CP "0"
C379 C8        RET Z
C37A FDCB016E  BIT 5,(IY+1)
C37E 28D8      JR Z,TLOOP
C380 FDCB01AE  RES 5,(IY+1)
; BASE ADDR OF CHAR SET
C384 21003C    LD HL,3D00H-100H
C387 110800    LD DE,B
C38A C5        PUSH BC
C38B 47        LD B,A
C38C 19        CHL ADD HL,DE
C38D 1000FD    DJNZ CHL
C38F C1        POP BC
C390 EB        EX DE,HL
; NOW OUTPUT THE CHARACTER
C391 CDA6C3    CALL OUTVDU
; AND ADVANCE THE CURSOR
C394 0C        INC C
C395 79        LD A,C
C396 FE20      CP 32
C398 208E      JR NZ,TLOOP
C39A 0E00      LD C,0
C39C 04        INC B
C39D 78        LD A,B
C39E FE18      CP 24
C3A0 2086      JR NZ,TLOOP
C3A2 0600      LD B,0
C3A4 18B2      JR TLOOP

;ENTER WITH DE POINTING TO
;FIRST BYTE OF CHARACTER CELL
;AND WITH BC COORDINATES SET
;TO THE CURRENT CURSOR POSITION
C3A6 C5        OUTVDU PUSH BC
C3A7 CDB8C3    CALL CONVERT
C3AA 0608      LD B,B
C3AC 1A        LOOP4 LD A,(DE)
C3AD 77        INC H
C3AE 24        INC DE
C3AF 13        DJNZ LOOP4
C3B0 10FA      ;SMALL DELAY
C3B2 0600      LD B,0
C3B4 10FE      D1 DJNZ D1
C3B6 C1        ;POP BC
C3B7 C9        RET
C3B8 C5        CONVERT PUSH BC
C3B9 78        LD A,B
C3BA E618      AND 18H
C3BC C640      ADD A,40H
C3BE 67        LD H,A
C3BF 78        LD A,B
C3C0 E607      AND 7
C3C2 0F        RRCA
C3C3 0F        RRCA
C3C4 0F        RRCA
C3C5 81        ADD A,C
C3C6 6F        LD L,A
C3C7 C1        POP BC
C3C8 C9        RET
C3C9 010204080
C3CD 10204080 UDG  DB 1,2,4,8,16,32,64,128
C3D1 FFFFFFFF CURSHP DB 255,255,255,255
C3D5 FFFFFFFF DB 255,255,255,255
C3D9 00000000
C3DD 00000000 NOCUR DB 0,0,0,0,0,0,0,0
C3E1 00        COUNT DB 0
C3E2 00        CHIGH DB 0
END

Workarea - A50A to A6BC
ORG end - C3E3
LOAD end - C350

```

The cursor-blink speed also depends on the length of code in the TLOOP loop, but that can't easily be adjusted.

C3D1: This holds the cursor shape, CURSHP, a black square of 64 inked pixels. UDG above it illustrates the code for a different shape, a diagonal line. The cursor shape is merely a user-defined graphic and can be designed on graph paper, or by using the UDG-generator supplied on the Spectrum "Welcome" tape.

C37A: The Sinclair ROM sets the Z80 register IY to point at address 23610 and IY+1 points at one of the FLAG locations. Bit 5 of this flag is set whenever a keypress has been detected (and loaded into 23560) by the interrupt routine. If BIT 5 is zero, no further action is taken and the program flow loops back into TLOOP.

C380: Having found a new keypress in 23560, BIT 5 of IY+1 must now be reset to zero, otherwise the screen will quickly fill

with the same character – an illegal auto-repeat mode!

C384: HL is set to point to the massive shape-table holding the pixel patterns for the Spectrum character-set. Each character shape is stored in 8 bytes of code, but

“... pre-blown EPROMs and PCBs will be available making construction easy ...”

there are no patterns provided for the 32 ASCII codes 00 to 1FH, which is why the mysterious 100H (32*8) is subtracted from the table start at 3D00H. The shadow EPROM will have symbols for codes 00-1FH added, since this will assist greatly in creating the embedded codes which program do-matrix printers.

C3AC: The code here transfers the pixel pattern from the character shape table to register A and hence (at C3AD) to the VIDEO RAM. HL holds the VIDEO RAM address for the current cursor co-ordinates, as computed by the CONVERT routine at C3B8.

C3B2: This delay simulates the length of time it would take to refresh the 17 x 32 screen window in the wordprocessor proper. Without it the Keyboard-Road/VDU-Write sequence is so fast that the 2mS interrupt routine begins to fail to detect the occasional key-press and spurious characters appear on the screen.

COMING NEXT

Next month the workings of the wordprocessor itself are dealt with and the package, in its shortened form, can become operational.

A serious side to Amstrad

Michael Graham previews the '664

The Amstrad Plus, more correctly the Amstrad CPC664, should be in the shops by the time this review appears in print. The new computer can best be summed up as a '464 with an in built 3" disk drive replacing the old 'datacorder'.

News that Amstrad were planning such an upgrade has been repeatedly leaked in the specialist and national press and so the CPC664 will come as no great surprise to the buying public. Given these facts, the most newsworthy item to come out of the launch was confirmation of the retail prices of the new computer. As with its baby brother, the '664 will be sold together with either a green screen monitor, in which case the price will be £339 or with an RGB monitor at £449.

Both systems will come with the CP/M operating system allowing a wide range of application software to be used in conjunction with the machine. A second disk drive, which makes many software packages far easier to use, will also be available at a cost £159. These prices are some £100 above the cost of the equivalent '464 packages but represent a useful saving on the purchase of an add-on disk drive for this machine.

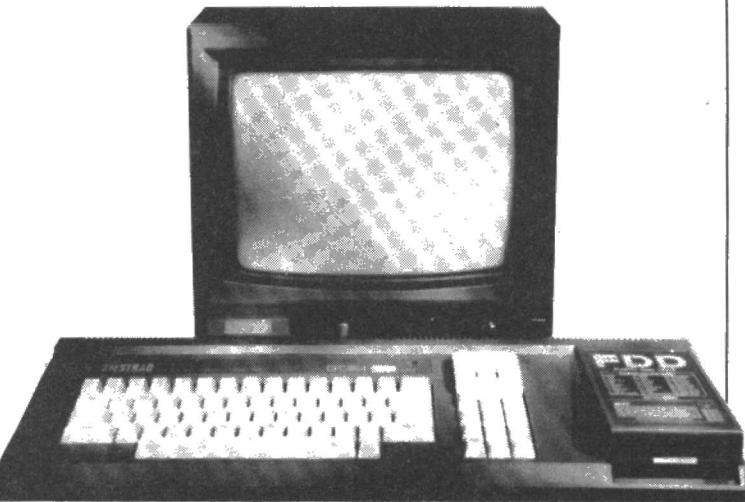
Apart from the addition of the disk drive Amstrad have decided to improve certain other aspects of the earlier computer's design. The most obvious are the changes made to the '664's case. Some changes to the moulding were obviously necessary in order to house the long, thin disk drive. Others were based upon suggestions made by users of

the '464. These include the provision of an MSX style cursor key cluster and a relegending of the numeric keypad. The numeric keys of the earlier machine could be used as function keys but this fact was often overlooked. To reinforce this role the keys on the new computer are labelled f0-f9.

BASIC enhancements

The other major difference between the two Amstrad models is that the newer machine has an enhanced, version 1.1, of Locomotive BASIC. The new commands are shown in **Table 1**. From this it can be deduced that the majority are concerned with the way in which the '664 handles itself when producing graphics. Other firmware enhancements include improvements of the way in which the system handles errors. This is the recognition of the fact that unlike a tape storage system in which errors are generally fatal, disk systems can generate a number of non-fatal errors and the programmer should be provided with the means to deal with these as appropriate.

The last point to note about the CPC664 is that it has grown, it has got an extra I/O port. No prizes for guessing that this is a DIN connector allowing tape-based software to be loaded into the computer. Amsoft will be providing a tape to disk transfer service that will allow users of the computer to send their cassette based software to the company and, for the price of a 3" disk, have it copied to the disk.



In perspective

That then is the CPC664 described in brief; how does it fit into Amstrad's overall marketing plan and into the home computer market in general? There can be no doubt that the new computer successfully builds on the generally well accepted, and commercially successful design of the earlier machine. In incorporating the disk drive in the new model, Amstrad still have a product with mass appeal although the new hardware has taken the company into the up market area of micro computer sales. The fact that CP/M is bundled in with the '664 means that users will have access to a vast range of software that should meet the needs of many groups of users, including that ever present small businessman.

Amstrad will continue to sell the CPC464 and this is seen as the ideal first time buyer's computer. An advertising campaign to promote the '464 is planned for the Autumn and it is this computer rather than the '664 which the company sees as taking the major share of the Christmas market. Amstrad plan to ship 600,000 computers in the current year and although they refused to comment on the split between 464 and 664, it is likely that the 464 will take in excess of two thirds of the total.

One final point to make is that version 1.1 of the BASIC will not find its way into the CPC464. The reason for this is that some third party software houses have produced games with code that doesn't follow the rules laid down in Amstrad's firmware guide. As a result not

TABLE 1. New BASIC commands

CLEAR INPUT
COPYCHR\$
CURSOR
DEC\$
DERR
FILL
FRAME
GRAPHICS PEN
GRAPHICS PAPER
ON BREAK CONT

Enhanced commands
AUTO
DRAW
DRAWR
MOVE
MOVER
PEN
PLOT
PLOTR
PRINT USING

all software will run under the new version of BASIC and Amstrad is playing it safe by sticking to version 1 or the BASIC interpreter as far as the '464 is concerned. The company chairman, Alan Sugar, did however state that computer 'buffs', among the ranks of which are many *E&CM* readers, will be given sympathetic consideration if they apply to the company for a new version of the ROM.

With so many computer companies having a rough time at the moment it is refreshing to see at least one concern still positively committed to the production of microcomputers and confident that it will be making money in the process.

Amstrad has shown that with the right management and imaginative marketing and packaging, it is still possible to succeed in the micro market.

NET

NEWS

Compiled by Sid Smith. Robert Schifreen will be back next month.

● Acorn has joined the list of computer manufacturers moving into viewdata services with the opening of a database of product information for dealers and owners.

The system runs on three BBC Micros and a Winchester hard disk, linked by Acorn's proprietary Econet local area network.

Lawrence Hardwick, Acorn's head of Econet developments, sees the system as serving to field the 80% of product enquiries which are totally routine.

"We still get asked about the price of the BBC Micro," he told us incredulously – apparently forgetting that the chaotic state of Acorn's price structure makes any such enquiry totally reasonable. A less-publicised reason for the service is the post-share-crisis redundancies in Acorn's Customer Services Department.

Anyhow, should you require to know the current state of some long-promised Acorn peripheral at 3am in the morning, full details will be found on Cambridge (0223) 243642. But don't all ring at once – the system only has three incoming lines.

● Micronet 800, the Prestel-based computer magazine, is introducing a do-it-yourself facility for amateur information providers.

For 25p a shot, Micronet subscribers can rent up to 27 Prestel frames for their exclusive use.

From June 1st, Micronetters will be able to compose Prestel pages on their own micros, and then send each page to Micronet in the form of a response frame – Prestel's equivalent of a pre-paid envelope. These will be picked up by Micronet and published

on the system as part of a once-daily bulk editing sweep.

Micronet has also announced an elaboration of its CB radio emulating Chatline facility – a service which enables members to send electronic mail for immediate publication on the system.

The magazine aims to overcome the present crowding of Chatline (and its monopolisation by drivelling adolescents) by transferring the system to a new, Prestel-supplied mainframe computer.

Instead of the present 15 or 20 minute wait before the appearance of a message, Micronet reckons that its public display will be instantaneous. And the new Chatline will be split into half a dozen special interest groups, ending its domination by the loudest and lowest.

Chatline is already a success – registering either first or second most accessed area in the monthly Prestel survey, and publishing around 400 messages per evening. The promised upgrade, due in the autumn, should give the service an excellence to match its popularity.

However, these two desirable additions to the Micronet database just happen to coincide with a price rise. The quarterly charge for the Prestel Microcomputing Closed User Group, of which Micronet is part, has risen by two pounds – or, as Micronet likes to phrase it, by 15p per week.

● In what promises to be a crucial test case, two comms users have been charged with an offence arising from alleged misbehaviour on Prestel.

After a three month investigation, freelance journalists Steve Gold and Robert Schifreen have been accused with "forging an instrument, namely a computer disk" under the 1981 Forgery and Counterfeiting Act.

The two men were arrested following a late-night swoop on their homes in Sheffield and North London by police and British Telecom officers. Computers, modems, disks, tapes and documents were taken in the raids, and the men later transferred to Fraud Squad headquarters in Holborn, London, where they were charged.

According to legal experts, the British law of forgery relies on the concept of a document which lies about itself – usually by purporting to originate from someone other than the true source.

In the case of the two arrests, the document (or "instrument") is a hard disk on the Prestel computer, and the charges are based on the idea that anyone who accesses or alters such a disk, while pretending to be someone else, has committed a forgery.

At the time of writing, Steve Gold (who writes Micronet 800's daily Micromouse column) and Robert Schifreen (who usually writes these pages) are on unconditional bail awaiting a second appearance in Bow St Magistrates Court.

In an equally significant case, an 18-year-old has been convicted of malicious damage after erasing programs stored on his employer's computer.

Angered by his boss's attitude in an overtime dispute, the youth caused damage which held up production for four days and took 80 hours to put right.

Along with 140 days of community service, the youngster has earned himself a place in the law books – the case is the first in which the concept of malicious damage has been extended to computer files.

● The biggest gap in comms equipment for business computers has been plugged with the release of a BABT-approved internal modem for the IBM PC.

The aptly-named Missing Link modem, developed by the ex-Torch duo of Colin Alton and Phil Rushby, costs £399 + VAT and is already attracting the attentions of software houses – Lotus are allegedly writing material to enable the modem to automatically retrieve information from an on-line database and incorporate it into stand-alone IBM software.

● Britain's largest online news service, Thorn EMI's Datasolve, has now added the Financial Times to its list of news sources.

The FT joins other Datasolve news providers like the BBC Summary of World Broadcasts, The Washington Post, The Economist, The Guardian and

Japan's Asahi News Service in supplying instantly-available information.

The FT won't be available from the moment it hits the streets – Datasolve can only promise availability "within 48 of publication" – and back issues only extend as far as January 2nd 1985.

However, Datasolve's 200 million word database does have some very fancy (but easy to learn) keyword search facilities, enabling selection according to a whole range of parameters, so whatever information is possessed is highly accessible.

Meanwhile, the £1 per minute database, located at Sunbury near London, claims to be negotiating joint deals with manufacturers of terminals desk-top computers, on the grounds that, "their customers are our customers".

Micronet Modem

Top 10

Compiled March/April

1. Prism 1000
2. Prism VTX5000
3. Commodore Communications
4. Pace Nightingale
5. Acorn Prestel
6. OEL Commodore Comms Pack
7. Wren Portable Computer
8. OEL Telemod 2
9. Tandata TM1-10
10. Miracle Technology WS2000

Micronet 800 –

Telesoftware Top

Three

Compiled March/April 1985

ZX Spectrum

1. 180 – CCS
2. Syphax – K. Alston
3. Tascopy Monochrome – Tasman

Commodore 64

1. Slurpy – Creative Sparks
2. Typing Wizard – Severn Software
3. Aztec Tomb Adventure – Alligata

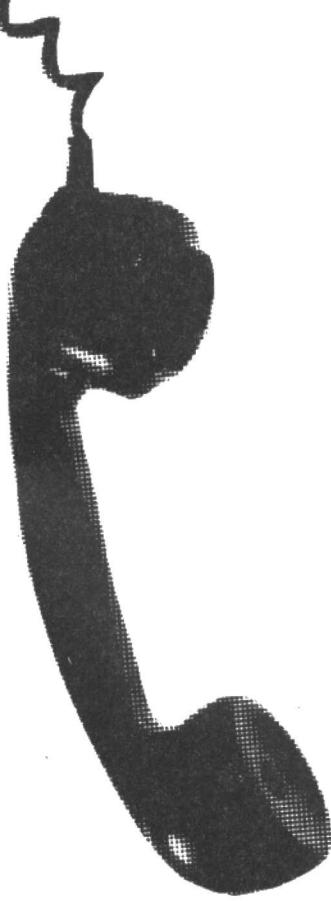
BBC Model B

1. Treepo – K. Campbell
2. Air Attack – M. Pryka Simpson
3. Stock Car – Micropower

NET

QUESTIONS

Distraught comms users tell all to the agony aunt of the networks.



Q: I'm writing Prestel software for my BBC Micro and can't get it to work. There is some recognisable text on the screen but no colour and lots of rubbish. What am I doing wrong?

A: Your difficulties arise from the fact that the BBC's implementation of the Broadcast Teletext Specification differs from Prestel's. Prestel, and all other viewdata systems, provide a 7-bit environment, that is, the character coding is based upon bytes of seven bits plus a parity bit to allow for error detection. The BBC

Mode 7 has no need of error detection and so uses an eight-bit version of the specification (see p488 of the User Manual).

In order to switch between, say, alphanumerics and alphanumerics, or to change foreground or background colour, Prestel sends to its terminals a pair of characters called a display modifier (because it modifies the display on the rest of the line). The first of the two characters is ESCAPE (ASCII 27). So, for instance, the Prestel display modifier for ALPHANUMERICS YELLOW is ESC C (ASCII 27,67). If you VDU 27,67 to the MODE 7 screen the 27 will be ignored and just "C" will be printed.

Fortunately, the BBC's coding of these display modifiers makes it easy to convert characters from Prestel into their MODE 7 equivalents. When you receive an ESCAPE from Prestel, don't print it, just add decimal 64 to the ASCII value of the next character to arrive and print that to the screen.

In BASIC you would write:

```
c% = FNnextchar
IF c% = 27 THEN VDU c%
FNnextchar + 64 ELSE VDU c%
```

where FNnextchar is the function which returns the ASCII value of the next character to arrive on the RS423.

Q: When I display Prestel frames containing double-height characters using my own terminal software on the Beeb only the top half is printed and sometimes I get some text appearing on the line below. Why is this and how do I cure it?

A: The double-height modifier of the BBC Micro's MODE 7 has a different effect to that on Prestel and this is the BBC Micro's only significant departure from the Broadcast Teletext Specification. On the BBC Micro you need to duplicate on the line below all characters which are to be printed in double-height. For example, to produce HELLO in double height on the Beeb you print:

```
dh HELLO
dh HELLO
```

where dh is the double-height modifier ASCII 141. On Prestel this duplication is not required so that:

```
dh HELLO
```

where dh is the Prestel double-height modifier ESC M (ASCII 27,77) has the same effect on a true Prestel terminal. But for this small anomaly Prestel screen handling on the Beeb would be very simple.

When you receive ESCAPE M from Prestel delete the line below, VDU 141 in the current cursor position and in THE POSITION DIRECTORY BELOW then VDU each subsequent printable character and modifier at the cursor AND in the position directly below. Do this until

you a) reach the start of the next line or b) receive a single height modifier, ie ESCAPE L.

So, upon receiving the double-height modifier:

```
x=POS:y=VPOS
VDU 13,10:PRINT STRING$(" ",40);:VDU 31,x,y
VDU 141,8,10,141,11
```

then for each printable character or modifier c%:

```
VDU c%,8,10,c%,11
```

until either of the above conditions are met. But you must remember afterwards NOT to display printable characters on the lower line – just advance the cursor (VDU 9) instead. This may sound a little long-winded but the Beeb can easily keep up without your having to resort to machine code.

Even though the above solution is adequate for ordinary Prestel frames it will not treat dynamic frames correctly. More of that later.

Q: I want to use my Prestel set instead of a VDU to access my mainframe at work. Can I do it?

A: It's possible, but the results are far from desirable.

Even if you can fix up a 75 baud receive/1200 baud transmit auto-answer modem onto one of your mainframe ports it will probably be expecting a conventional 72 or 80-character scrolling visual display unit as a terminal. Your Prestel screen is 40 characters wide and can't be persuaded to scroll! The result will be words broken over line boundaries making a mess of any tabulations. Also, after printing text on the bottom line your Prestel set will simply reposition the cursor at the top of the screen and continue printing from there. This means that old text will be left on the screen and partially overwritten by new text, making it barely readable.

Of course, you could write a view-data presentation converter for your mainframe which would at least stop transmission to the terminal at appropriate points and wait for you to press a key before clearing your screen and continuing.

But designing such converters is not a trivial matter and involves some pretty nifty use of input and output buffers so, unless you're really determined, forget it and buy a VDU.

Q: My old CP/M micro will not print Prestel graphics but I have some terminal software for bulletin boards. Can I use this for Prestel if I get the right modem?

A: You can try but there may be problems with Prestel's display modifiers (see above) and your screen width.

Your terminal software may do strange things with the ESCAPE character and you must trap this to

avoid trouble. When you receive an ESCAPE just print a SPACE on the screen (ASCII 32) then wait for the next character and DON'T print it. If it's Q R S T U or V then print an asterisk for each subsequent printable character unless it's an uppercase letter. Ignore any other character immediately following ESCAPE.

Remember that Prestel has 40 character lines so after receiving 40 printable characters you must print carriage-return line-feed (ASCII 13,10) to reposition the cursor – Prestel won't do it for you. Other characters to watch are HOME (ASCII 30) and CLEAR SCREEN (ASCII 12).

Q: What are 'dynamic frames' on Prestel and how do I change my Beeb terminal software to cope with them?

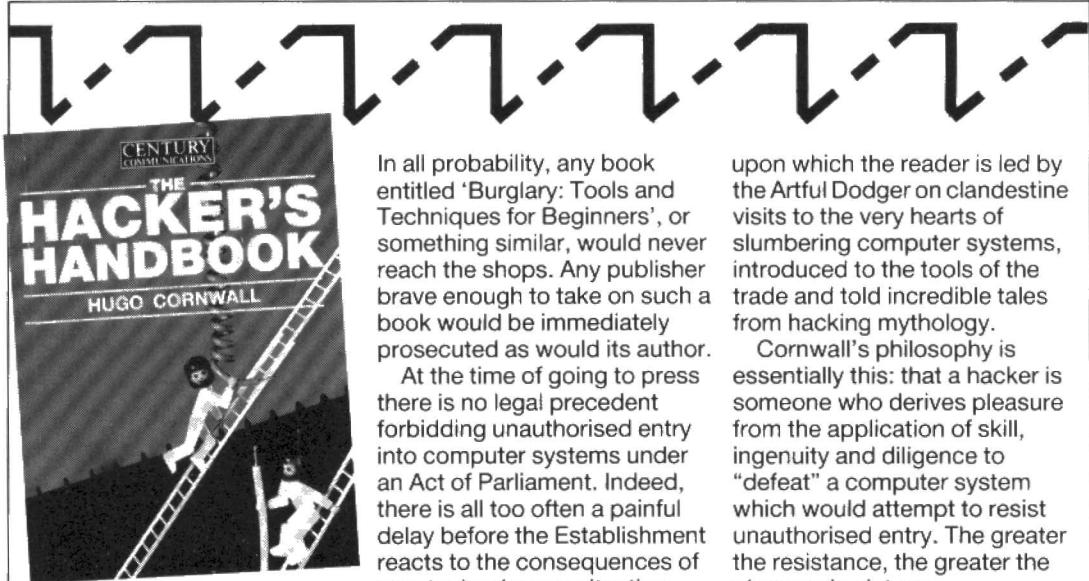
A: Frames on Prestel are usually sent to terminals as a continuous stream of up to 960 characters. These characters consist of printable characters, display modifiers and carriage-return line-feeds (CRLFs). CRLFs are only sent on lines shorter than 38 printable characters. DYNAMIC frames may be longer than 960 characters and any cursor control characters can be interleaved with other characters. The overprinting which this cursor movement allows means that animated displays are possible.

If you've BOUGHT your Prestel software it should be able to display dynamic frames as well as any other type (if it doesn't take it back and complain). But if you've written it yourself you may have to change radically the way you've been doing your screen display.

It is wise to have a general display routine which copes with any type of frame, dynamic or not. Instead of simply sending all received characters to the screen (after converting display modifiers) you must maintain a SCREEN BUFFER – an off-screen 40 x 24 copy showing the positions of the printable characters and modifiers you've received since the last CLEAR SCREEN character (ASCII 12).

You then use this buffer to 'refresh' the screen using the algorithm from Question 2 (above) as if the characters had just arrived. Remember that the buffer will differ from the screen whenever double-height (DH) or single-height (SH) modifiers are received because of the MODE 7 anomaly.

Cursor control characters can be printed directly to the screen. However, when you receive a DH or SH modifier you must refresh the whole screen from the cursor position onwards. You must also refresh the screen if you overprint a DH or SH modifier with another printable character or modifier. Other characters and modifiers must be copied to the buffer and to the screen.



The Hacker's Handbook

Hugo Cornwall
Century Communications
£4.95 Paperback 149pp

A review by Mike Brown, Technical Manager of Micronet 800. Any other opinions on the morality or immorality of hacking would be welcomed. Send your letters to The Editor, E&CM, Priory Court, 30-32 Farringdon Lane, London EC1R 3AU.

In all probability, any book entitled 'Burglary: Tools and Techniques for Beginners', or something similar, would never reach the shops. Any publisher brave enough to take on such a book would be immediately prosecuted as would its author.

At the time of going to press there is no legal precedent forbidding unauthorised entry into computer systems under an Act of Parliament. Indeed, there is all too often a painful delay before the Establishment reacts to the consequences of new technology, a situation which will worsen as the pace of innovation accelerates.

It is largely as a result of this inertia that 'The Hacker's Handbook' is enjoying a modest success for Century Communications and the opportunist for whom 'Hugo Cornwall' provides a flimsy anonymity. Current criminal proceedings under the Forgery and Counterfeiting Act 1981 could bring an abrupt end to this honeymoon.

The handbook amounts to a kind of voyage of adventure

upon which the reader is led by the Artful Dodger on clandestine visits to the very hearts of slumbering computer systems, introduced to the tools of the trade and told incredible tales from hacking mythology.

Cornwall's philosophy is essentially this: that a hacker is someone who derives pleasure from the application of skill, ingenuity and diligence to "defeat" a computer system which would attempt to resist unauthorised entry. The greater the resistance, the greater the pleasure in victory.

Fraud and vandalism, he says, are outside of the hacker's code of ethics; hacking is, as it were, burglary without theft.

The suggestion that ethical issues form any major component of the hacker's mentality is little more than a journalistic invention. He takes no account of the damage to the credibility of subscription networks caused by merely impish trespass; legitimate users could challenge with some justification a bill received

NET

OPINION

following a major security breach at system manager level.

Examples are cited of hackers helping to develop secure systems in America and there is mention of their reasserting the values of freedom, individuality and human worth. And I dare say some hackers have even been known to help old ladies with their teleshopping.

In short, Cornwall peddles the crass libertarianism of all those who earn a comfortable living encouraging others to indulge in legally dubious activity.

But not, perhaps for much longer.

BOOK REVIEW

The book is divided into five main sections, the first of which provides a general introduction to the attitudes and disciplines required when working with electronic equipment. This includes a list of the tools required if any modifications are to be made to the micro and an explanation of the art of soldering.

The major part of the book is taken up with a description of the operation of the micro. Each element of the system is described in considerable detail and in an informative and easy to understand fashion. One problem faced by the publishers was that Acorn refused permission to publish the circuit diagram of the computer. To get the most from the book it would be as well to try and track down such a diagram (E&CM published the main PCB's circuit in its April issue).

Having explained the operation of the computer, the book moves on to explain the function of the numerous links found on the main PCB. These control many fundamental characteristics of the machine's operation including the configuration of the printer interface and the disk controller.

A further section explains how the computer can be upgraded by the user; this includes explanations of how to install a floppy disk interface, a speech synthesiser and an Econet networking interface.

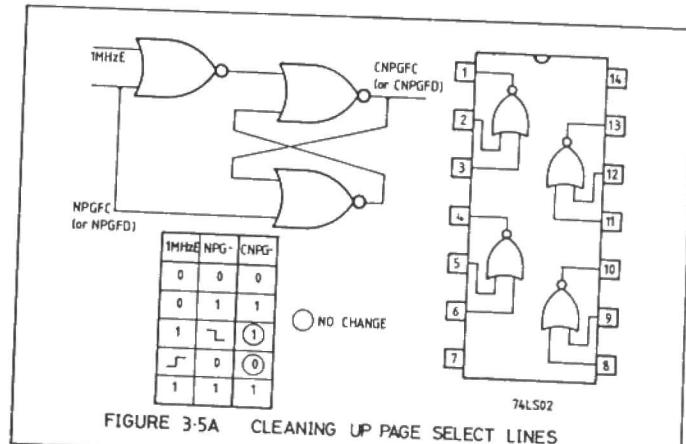
There is also a useful hints and tips chapter which aims both to pro-

vide cures to problems that crop up on a quite a few models of the computer, such as reliability problems with both the cassette and video interfaces, and to explain ways in which operation of the standard machine can be improved.

A significant section of the book is taken up with a very useful compendium of the data sheets associated with the various ICs within the computer. This section provides information not only on the major compo-

nents such as the VIA and MPU but also on the large number of TTL MSI support ICs.

This Hardware Guide should serve the needs of a wide range of BBC micro owners from the electronic tyro seeking to gain a basic insight into the operation of the computer, to the relative expert requiring a detailed explanation of the system, to modify and expand the computer to meet their own specific needs.



One of the many diagrams to be found in the guide.

BBC COMPACT FORMAT

This disk utility by Ian Ellerington allows the BBC user to double the number of files on each disk.

One of the major disadvantages of the Acorn DFS is that only 31 files can be stored on one disk. This program works on both 40 and 80 track disks and allows the user to have up to 58 files on one disk.

This is achieved by having dual directories, one stored on the first two sectors of the disk and the other stored on the second two sectors. A swap program is saved so both directories can be accessed simply by typing *RUN SWAP or just SWAP. To format a disk so that it may have 58 files, just type in the program, save it, run the program and insert a blank formatted disk into drive 0. Now press the space bar and the disk drive will start up; the computer will print the number of tracks on the disk and will format it with 58 files. If any disk errors occur during the program it will print the error and the program will stop. If this happens check the program and if no mistakes are found then the disk is faulty, so it must be reformatted. After the program has been run the catalogue of the disk should look like this:

```
(02)
Drive 0      Option 0 (off)
Directory:0.$  Library:0.$
SWAP L
Z.ZZZZZZ L
```

If the catalogue is different to this then you have made a mistake when entering the program – find it and correct it.

If you compact the disk it will have only half the normal number of sectors free. The other half can be accessed from the other directory. By typing *SWAP you will switch to the other directory, which will be shown by entering *CAT. Both directories will be exactly the same until you have saved a program on one of them. To test the program, do the following:

1. Save the setup program on the disk.
2. Change to the other directory by typing **SWAP'.
3. Check that the program you have saved is NOT on this directory by typing *'. If it is then there is a mistake in the program so check it out.
4. Finally type *SWAP to be sure the program is still on the original directory.

DO NOT delete either the 'SWAP' program or the file 'Z.ZZZZZZ', otherwise you will not be able to get to any program onto the other directory.

You now have a disk which can access 58 files.

LISTING: Dual Directory Software

```
>1
10 REM      SETUP PROGRAM TO HAVE 58 FILES ON ONE BBC DISC
20
30 REM      WRITTEN BY IAN ELLERINGTON JULY 1984
40
50 MODE 7
60 PROCsetup
70 PROCwrite
80 PROCsav
90 END
100
110
120
130 DEFFPROCsetup
140 #DRIVE 0
150 PRINT"CHR$129;" WARNING DESTROYS ALL PROGRAMS ON DISC"
160 PRINT"INSERT A FORMATTED DISC INTO DRIVE 0      AND PRESS THE SPACE BAR":REPEAT UNTIL INKEY(0)=32
170 VDU21:.*
180 VDU6
190 hitracks=?&F06 AND 31otracks=?&F07
200 tracks=(hitracks*256+otracks)/10
210 IF tracks <>40 AND tracks <>80 THEN PRINT"CHR$129;" BAD DISC SIZE":END
220 PRINT"CHR$129;tracks;" TRACK DISC"
230 PROCsctors
240 PROCtable
250 ENDPROC
260
270
280 DEFFPROCassembly
290 FORNX=0 TO 2 STEP2
300  P%=&C00
310  COPTNX
320  JSR oswrd:BEQ label
330  JMP error
340  .label INCsector:INCsector:INCaddress
350  JSR oswrd:BEQ label2
360  JMP error
370  .label2 LDA#4B
380  STAcommand
390  DECsector:DECsector
400  JSR oswrd:BEQ label3
410  JMP error
```

```
420  .label3 INCsector:INCsector:DECaddress
430  JSR oswrd:BNE error
440  LDA#53
450  STAcommand
460  DECsector
470  JSR oswrd:BNE error
480  INCsector:INCsector:INCaddress
490  JSR oswrd:BNE error
500  DECsector:DECsector
510  LDA#4B:STA command
520  JSR oswrd:BNE error
530  DECaddress
540  INCsector:INCsector
550  JSR oswrd:BNE error
560  LDA#E:STAaddress
570  INCnes
580  LDA#53:STAcommand
590  LDA#0:STAsector
600  JSR oswrd:BNE error
610  RTS
620  .error
630  PHA
640  LDX #10
650  .loop LDA data.X
660  JSR &FFEE
670  DEX
680  BPL loop
690  PLA
700  PHA
710  LSR A
720  LSR A
730  LSR A
740  LSR A
750  JSR NIBBLE
760  PLA
770  AND #F
780  JSR NIBBLE
790  LDA#10:JSR&FFEE
800  LDA#13:JSR&FFEE
810  RTS
820  .NIBBLE
830  CLC:ADC #4B
840  CMP #5B
850  BMI print
860  CLC:ADC #7
870  .print JSR&FFEE:RTS
880  .data :1
890  *P%=" RORRE CSID"
900  P%+P%+12
910  COPTNX
920  .oswrd LDA#7F:LDX#&DB:LDY#&C:JSR&FFF1
930  LDA err
940  RTS
950  J
960  RESTORE 1200
970  FOR X=P% TO P%+9:READ A: ?X=A:NEXT
980  err=P%+10
990  command=P%+6
1000  address=P%+2
1010  sector=P%+8
1020  nosp=P%+9
1030  NEXT
1040 ENDPROC
1050
1060
1070
1080 DEFPROCwrite
1090 A%=?F: X%=?Y%=&A: CALL&FFF1
1100 IF?&OA0=0 THEN ENDPROC
1110 PRINTCHR$129;"DISC ERROR &";?&OA0" AT 0/";?&OA0:END
1120
1130
1140 DEFFPROCsectors
1150 FORX=&1100 TO &14FF
1160  ?X=0
1170  ?&120F=2
1180  NEXT
1190 $&110B="ZZZZZZZ":?&110F=&DA
1200 $&130B="ZZZZZZZ":?&130F=&DA
1210 ?&1205=8
1220 ?&1206=?&F06
1230 ?&1207=?&F07
1240 IF tracks=40 ?&120D=&C8 ELSE !&120D=&1090
1250 IF tracks=40 ?&1407=&C8 ELSE !&1406=&9001
1260 ?&120F=2
1270 ?&140F=2
1280 ?&140D=2
1290 ?&1405=8
1300 ENDPROC
1310
1320
1330 DEFFPROCtable
1340 RESTORE 1520
1350 FORX=&A00 TO &A09
1360  READ A: ?X=A
1370  NEXT
1380 ENDPROC
1390
1400
1410 DEFPROCsave
1420 PROCsassembly
1430 *SAVE SWAP C00+E3
1440 *ACCESS SWAP L
1450 CALL&C00
1460 PROCsassembly
1470 *SAVE SWAP C00+E3
1480 *ACCESS SWAP L
1490 ENDPROC
1500
1510 DATA0,0,9,0,0,3,&53,0,0,&21,&FF
1520 DATA0,0,&11,0,0,3,&4B,0,0,&24,0
```

A new EMPHASIS

How to get better printer effects from Wordwise and other wordprocessors

An extract from the Wordwise Applications Guide, by Paul Beverley.

Not all printers have the same character set, especially daisy wheel printers which vary considerably from wheel to wheel. Variations in typeface cause no problems, but the lack of standardisation of symbols and punctuation marks can create considerable difficulties. This article sets out to solve some of those problems, with special reference to Computer Concepts' BBC wordprocessor, Wordwise, but the program could be used with other wordprocessors or from Basic.

Exclamation marks

Some daisy wheels, especially those for use with printers which were designed to work as type-writers as well (eg the Brother HR1 which I use) do not have an exclamation mark. A typist would be used to using a fullstop back-space + apostrophe, but word-processors are not normally programmed to cope with that sort of thing! To deal with this in Wordwise, you can use the exclamation mark as normal while editing the text, and then just before printing out the document, use a programmed key to make the necessary changes:

*KEY 7 !!\$!A.!!!OC8,39!!"

This moves the cursor to the next exclamation mark (:!\$!), deletes it (:A), replaces it with a full-stop and puts in an OC command which sends out a backspace followed by an apostrophe. (8 is the ASCII code for backspace and 39 for apostrophe.)

If you know that you are only ever going to want to print out the text on the daisy-wheel printer, you could save the file with these embedded commands. However, if you do so, you may find that it will not print out properly on a dot-matrix printer. Some printers are not capable of doing backspaces and others use a slanting apostrophe, more like an acute accent, which produces a very strange looking exclamation mark.

The alternative solution which we are going to look at later is to use a program which forces the computer to make the necessary changes automatically. It sends the extra characters to the printer whilst at the same time displaying the original exclamation mark in the edit and preview modes.

LISTING 1. HR1 printer effects (should work for most daisywheel printers)

```

10 REM Program to provide auto-underline on Brother HR1
20 REM with slashed zeros, and exclamation marks.
30 REM (c)1985 Norwich Computer Services
40 PROCsetup variables
50 PROCassemble(&C00)
60 PRINT" *SAVE HR1 ";"start" ";"P%" ";"start"
70 CALL &C00 : REM Set it up for use now.
80 END
90
100 DEFPROCassemble(M%)
110 FOR opt=0 TO 2 STEP 2
120   P%-M%
130   [OPTopt
140
150 .start
160   SEI
170   LDA INSVEC      \ "Character being inserted
180   STA oldINSVEC  \ into a buffer" vector.
190   LDA #newINSVEC MOD 256
200   STA INSVEC
210   LDA INSVEC + 1
220   STA oldINSVEC + 1
230   LDA #newINSVEC DIV 256
240   STA INSVEC + 1
250
260   LDA WRCHVEC      \ Vector for OSWRCH
270   STA oldWRCHVEC
280   LDA #newWRCHVEC MOD 256
290   STA WRCHVEC
300   LDA WRCHVEC + 1
310   STA oldWRCHVEC + 1
320   LDA #newWRCHVEC DIV 256
330   STA WRCHVEC + 1
340
350   LDA #0           \ Start with underline off.
360   STA underline_ON
370   CLI
380
390   LDA #6           \ *FX6,0
400   LDX #0           \ i.e. send linefeeds.
410   JSR OSBYTE
420   RTS
430
440 .newINSVEC
450   CPX #3           \ Is it the printer buffer?
460   BNE jmpout        \ If not, ignore it.
470
480   CMP #0           \ If not a zero,
490   BNE jmpout        \ ignore it.
500
510 .change_flag
520   LDA underline_ON \ If it was a zero,
530   EOR #1           \ change state of the
540   STA underline_ON \ underline flag.
550   LDA #0           \ Restore zero ready to

```

A slashed zero

If you are writing a document in which a clear distinction must be made between zero and the letter O, you may on a daisy-wheel printer have problems. Not many daisy-wheels have the slashed zero. One possible solution is to use a similar technique to that for the do-it-yourself exclamation mark: simply add a back-space and slash symbol to each zero. This can be done by going through the document with a function key programmed as:

*KEY 7 !!\$0!-!!OC8,47!!"

This moves to the zero (:!\$0), moves past it (:!-) and puts in the OC codes for back-space and slash. However, you may find that the resulting character looks rather strange, particularly if the zero is a less rounded character than the capital O. In which case you could try:

*KEY 7 !!\$0!AO!!OC8,47!!"

This moves to the zero (:!\$0), deletes it (:A), prints a capital O instead, and then adds the slash character on top.

As with the exclamation mark problem, it is possible to get the computer to do this for you automatically by using a machine code program.

Super & sub-scripts

Some dot-matrix printers like the Epson FX80 have automatic super and subscripting. It is simply a case of looking in the printer manual to find out which OC codes to use to switch each facility on and off.

If you have a daisy-wheel printer then although you cannot produce different sized characters, if it is capable of executing forward and reverse half linefeeds, these can be used to produce super and sub-scripts. On the Brother HR1, to produce X² or H²O you can use:

X<gr>OC27,68<wh>2<gr>OC27,
85<wh>
H<gr>OC27,85<wh>2<gr>OC27,
68<wh>O

<gr> and <wh> are the conventions used in the book for command start and command end, ie F1 and f2).

Auto-underlining

Many printers these days have built-in intelligence and can offer facilities such as auto-underlining where you send a particular code or code sequence to the printer and thereafter it underlines every single character automatically until it receives another code to tell it to stop underlining. If you do not have this facility, there are a number of possible (partial) solutions.

For underlining headings, if you are NOT using auto-linefeed, you can try something like:

This is the title to be underlined

<gr>OC13<ret>

The idea is that this should send the title

LISTING 1 (Continued)

```

560                                \ send to the printer.
570 .jmpout
580     JMP (oldINSVEC)
590
600 .newWRCHVEC
610     STA Astore
620     STX Xstore
630     STY Ystore
640
650     CMP #2
660     BNE isprinteron
670
680     LDA #0
690     STA underline_ON
700
710 .isprinteron
720     LDA #75
730     JSR OSBYTE
740     TXA
750     AND #1
760     BEQ out
770
780     LDA Astore
790     CMP #32
800     BCC out
810
820     CMP #ASC("!")
830     BNE isitanought
840
850     LDY #ASC("''")
860     JSR sendYtoPbuffer
870     LDY #8
880     JSR sendYtoPbuffer
890     LDA #ASC(".")
900     STA Astore
910
920 .isitanought
930     CMP #ASC("Ø")
940     BNE U_linecheck
950
960     LDY #ASC("/")
970     JSR sendYtoPbuffer
980     LDY #8
990     JSR sendYtoPbuffer
1000    LDA #ASC("O")
1010    STA Astore
1020
1030 .U_linecheck
1040     LDA underline_ON
1050     BEQ out
1060
1070     LDY #ASC("_")
1080     JSR sendYtoPbuffer
1090     LDY #8
1100     JSR sendYtoPbuffer
1110
1120 .out
1130     LDX Xstore
1140     LDY Ystore
1150     LDA Astore
1160     JMP (oldWRCHVEC)
1170
1180 .sendYtoPbuffer
1190     STY Ystore2
1200
1210 .wait
1220     LDA #128
1230     LDX #&FC
1240     JSR OSBYTE
1250     CPX #2
1260     BCC wait

```

\ Save registers.

\ Is it a CTRL-B (VDU 2)?

\ Switch off underlining in case it was left switched on last time.

\ Check VDU status.

\ Is printer on?

\ If not, give up.

\ If it's a control code, give up.

\ Is it a "!"?

\ Send an apostrophe, then a backspace.

\ Put a full-stop ready to send to printer.

\ Is it a zero?

\ Add a slash, and a backspace.

\ Send upper case O instead of zero.

\ Is underline on?

\ Send the underline code and a backspace.

\ Restore the registers.

\ Check the printer buffer.

\ Is there at least one space left?

line to the printer and then a carriage return but not a linefeed. The print head should go back to the beginning of the line and overprint with the underline characters. This will not work if you are using the auto-linefeed mode because the OC13 command will cause the printer to generate the linefeed as well. It is important to see that we have used a <ret> rather than a <wh> after the command. This means that Wordwise sees the underlining characters as part of the same line – otherwise you would end up with one line short on the page. However, if the title is longer than half the normal line length, the formatting will still get confused, so for longer titles you will also have to increase the line length before the title and restore it to normal after the title.

If you need to underline words within the body of some text – because of the formatting – you can't use the method above since the position into which the word will appear is unknown. If you only have the odd word to underline here and there then you could do it by issuing OC commands for the requisite number of backspace and underline characters. Thus in the following example, to underline the word "sure" we would use:

Are you sure<gr>OC8,8,8,8,95,95,95,
95<wh>about that?

This obviously becomes unwieldy for words more than a few characters long.

Implementing and modifying the program

The program shown in Listing 1 was originally written for the Brother HR1 but has also been tested on the Juki 6100. It should work for most other daisy-wheel printers, although since it has not been tested on other printers this cannot be guaranteed.

What the program does is to create a machine code program and, once this program has been created and saved on tape or disk, it can be installed in the computer at any time. You do not have to go into BASIC to set it up. You can be in Wordwise and already have some text typed or loaded in, ready for printing.

To create the machine code program, all you have to do is to switch to BASIC, type the program in, check it through carefully and then RUN it. It prints out a line which says "SAVE HR1" and then some hieroglyphics. (You may want to change the file name, "HR1" at line 60, to "Juki" or whatever is an appropriate name.) This is the line that you should type in order to save the machine code program on tape or disk. As it is already printed out on the screen, you can enter it by using the cursor and copy keys. Note that this only saves the machine code program and not the BASIC program which created it. This is referred to as the "source code" and can be saved as any other BASIC program with SAVE "HR1SCE" or whatever you like to call it. In theory you no longer need this source code program, but it would be sensible to save it, just in case you need it again, or want to modify the program at a later date.

LISTING 1 (Continued)

```
1270      LDA #138          \ Osbyte 138, put a
1280      LDX #3             character into a buffer.
1290      LDY Ystore2
1310      JMP OSBYTE
1320
1330      .underline_ON      \ Save space for variables.
1340      NOP
1350      .Astore
1360      NOP
1370      .Xstore
1380      NOP
1390      .Ystore
1400      NOP
1410      .Ystore2
1420      NOP
1430      .oldWRCHVEC
1440      NOP:NOP
1450      .oldINSVEC
1460      NOP:NOP
1470      ]
1480      NEXT
1490      ENDPROC
1500
1510      DEF PROCsetup_variables
1520      INSVEC=&22A
1530      WRCHVEC=&20E
1540      OSBYTE=&FFF4
1550      ENDPROC
```

When you want to use the printer, if you have disk, you just use *HR1<return> and that automatically loads and runs the machine code program which sets up the BBC system to send out the necessary codes for auto underline, slashed zeros and exclamation marks. With a tape system you say *RUN<return> or just */<return>. The text itself does not need changing in order to produce the zeros or the exclamation marks, but to get the underline to switch on and off, you use OC zero. For example:

This is <gr>OC0<wh>underlined
<gr>OC0<wh>, but not this.

If you don't want slashed zeros, simply delete lines 930 to 1030, and if you don't want exclamation marks made up of full-stops and apostrophes then delete lines 820 to 920 inclusive. If you think you might want different combinations of these facilities depending on the conditions, why not RUN the program four times with and without the lines mentioned. (Remember to save the whole program first to avoid re-typing.) You can save them as four different files, HR1 (underline only), HR10 (with zeros), HR11 (with exclamation marks) and HR10! (with all features). The only thing to note is that you should disable one version before loading in an alternative one. You do this by pressing the break key and then pressing <Y> to get your old text back.

The slashed zeros are done by adding backspace and slash, but as it stands, the program also replaces the zero with a capital O as mentioned above. The reason for this is that it happens to look better with the particular daisy-wheel I use for listings. If you would rather just leave the original zero

and add the slash, simply delete lines 1000 and 1010.

The program as it stands includes the *FX6,0 command in machine code form. If your printer uses auto-linefeed, simply delete lines 390 – 410. If you have a serial printer, you could save yourself typing in *FX5,2 every time by including:

```
381  LDA #5
382  LDX #2
383  JSR OSBYTE
```

Generating accents for foreign languages

If you are using a foreign language that needs different accents, you can apply a similar technique to the one described above for exclamation marks or slashed zeros. I have not had cause to try this myself, but presumably if you have a daisy wheel that has the required accents somewhere on it, you could use appropriate OC commands to backspace and over-print the accent. If you want to do this automatically then the program just described could be modified to produce the appropriate codes. The section from 820 to 1010 could be extended to look for special characters which you want to use for the particular accented characters and then add the backspace and accents in a similar way to the zeros and exclamation marks. The only thing to be careful of is that you have to change the labels so that after each check you have a BNE to the next check forming a complete chain of checks for the characters to be modified. If you have any difficulties in implementing this, please contact Norwich Computer Services 0603 621157.

Turtle invaders

Adam Denning reviews
Acornsoft's Logo package

LOGO

BBC Model B

Acornsoft £69.95

Acornsoft LOGO comes as two 16K ROMs, a disk and a cassette. Accompanying this wealth of software are three manuals, a laminated reference card and a poster. You certainly get a lot for your money,

to MOVEALOT, ie what to do when someone types MOVEALOT into the computer. The END tells LOGO that the procedure definition has ended. Now, whenever the machine is told to MOVEALOT, it looks up its definition for that procedure and carries out all the commands therein. The embedded commands may be other procedures, they may be primitives

'A valiant attempt at simulating a simple Prolog front-end'

and most of it is very good indeed.

LOGO is of course the language of Seymour Papert, and this implementation follows his ideas pretty closely. LOGO consists of two distinct parts: the turtle graphics package and the programming language. Although Acornsoft released a turtle graphics program by itself a few months ago, it cannot hope to compete with this full LOGO.

A turtle is a notional being which has either the computer screen or the floor as its domain, and it can be lead around this area by simple commands such as FORWARD and RIGHT. The turtle can be made to draw lines as it moves, thus tracing its path, and it can change the colour of the line using other simple commands. Because of its immense visual impact and ease of instructive control, it is an ideal tool for teaching children mathematical concepts and 'spacial relationships'. As this invariably involves drawing complicated and pretty pictures, the children are kept happy as well.

To make a turtle do anything more than move from point to point drawing lines, it is more convenient to define procedures which contain a whole group of operations to be carried out upon invocation of the procedure. In keeping with the LOGO philosophy, a procedure is defined using the TO keyword, as in:

```
TO MOVEALOT
DOFOREVER [SQUARE]
END
```

Here the 'TO MOVEALOT' tells LOGO that it is about to be told how

(which are equivalent to built-in procedures), or they may be re-invocations of this procedure, so that recursion is implemented.

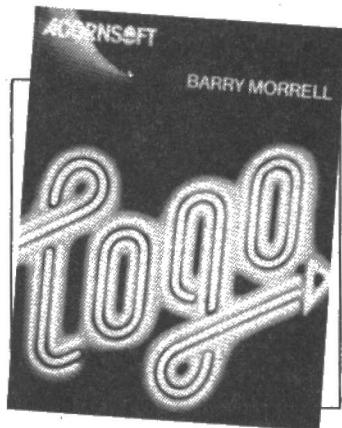
As you get more and more familiar with LOGO it is inevitable that you will be defining more and more procedures, and of course there are going to be times when you will want to alter a definition, either because it is wrong or because you have thought of a way of increasing the efficiency of the routine. These alt-

erations can be made by entering the LOGO editor, from where it is possible to edit any procedure or variable in the workspace. The editor is essentially a screen editor which works on a line at a time (if you see what I mean) and as it is built into the ROMs there are no worries about valuable workspace being taken up.

The final and more complicated aspect of LOGO is its little discussed list processing abilities; it comes across as a sort of cross between LISP and PROLOG. Just as in LISP, a LOGO list is a collection of objects and lists, where an object is a name or number, and a list is anything from the [] empty list to any number of delimited objects. Acornsoft LOGO supports all the usual list processing commands, and even includes an extension called LOGIC which makes a valiant attempt at simulating a simple PROLOG front end. As the manual says, this resembles true PROLOG about as much as a turtle graphics system resembles true LOGO, but it is nevertheless very worthwhile and great for demonstrating the potential power of the language. This version of LOGO was written in BCPL which partially explains its incredible size and power.

When both ROMs are installed and either the *LOGO command is issued or the machine is switched on, depending on ROM socket priority, LOGO is started. This causes the screen to go into a 40-column graphics mode and the turtle 'fence' to be drawn. This fence is an arbitrary out-of-bounds marker for the screen turtle, and commands which would drive the turtle over this boundary will generate the 'Turtle hit the fence' error, which is a far better message than, say, 'Out of range at line 100'.

The last six lines of the screen are devoted to text entry, and this is indicated by a '?' prompt. From this position, primitives such as FORWARD and LEFT can be entered, procedures can be defined and names can be set up. A name in LOGO is a variable in any other

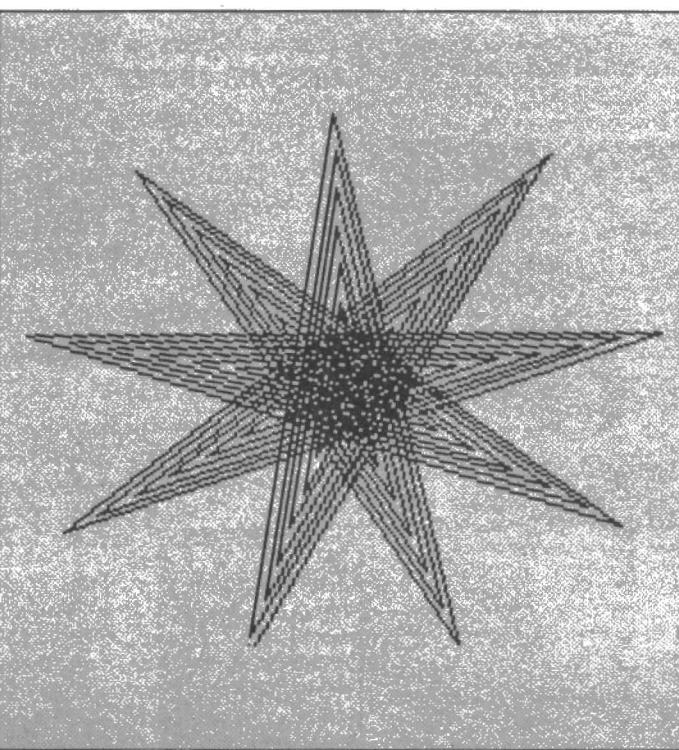


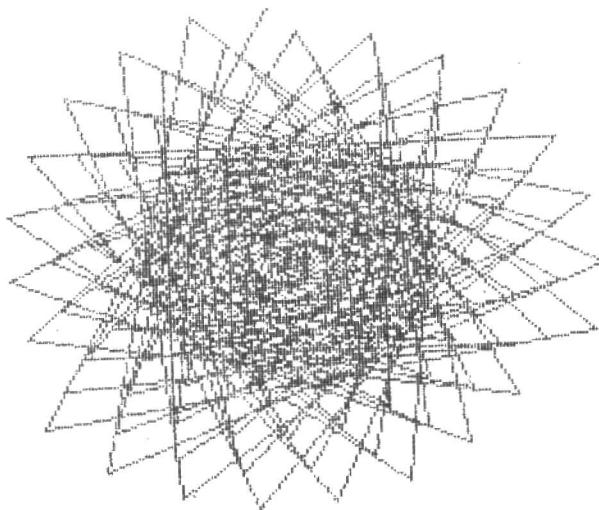
language. When a procedure is being defined the prompt turns into a '>' so that even if the text window scrolls the programmer will always know what state the system is in.

There is no denying that turtle graphics are very important to the LOGO concept, and so there are a great number of primitives defined for just this purpose. It is also possible to load one of the extensions from disk or cassette which will drive an external, floor bound turtle, such as the BBC Buggy, the Jessop turtle or the Valiant turtle. There is also the intriguing possibility of having up to 32 turtles on the screen at once, and each turtle can have a different screen appearance. This is easily alterable because each turtle's 'shell' is an ASCII character, and the default state a defined character representing a compass point. As the VDU command is accessible from within LOGO it is easy to redefine a character and use that as the turtle's shell instead. Could this mean the first LOGO space invaders?!

An intriguing aspect of this implementation is the way that the examples and extensions are supplied - as a 40 track disk (and cassette). This may seem to rule out owners of 80 track drives, but Acornsoft supplies a configuration program invoked by *CONFIG which backs up this disk onto an 80 track disk. As this utility seems to be a pure 40/80 backup program, it should be very useful in its own right, as the upgrade to an 80 track disk drive can often be the cause of many a headache with so many 40 track disks no longer being readable.

The three manuals supplied are written by Barry Morrell, who also wrote the manual for the earlier turtle graphics package, and who is apparently well respected in the computer educational field. He certainly knows how to write books

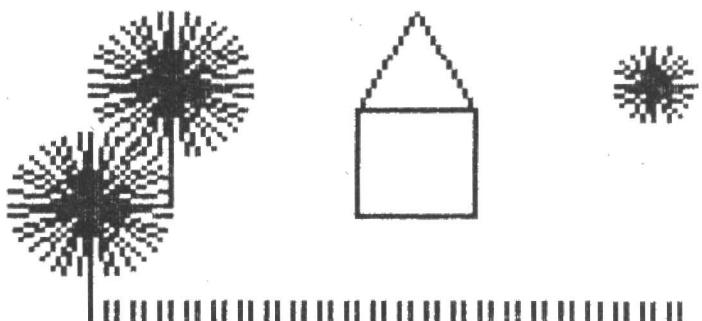




with a minus sign if it is a negative exponent, as in $1.2E-2$. As this can thoroughly confuse learners, LOGO replaces the whole of 'E-' by 'N'. Although this may seem a fairly trivial point, it does make things easier when you're dealing with the concepts for the first time. The LOGO arithmetic routines are all very fast and very accurate, being derived from Acornsoft's Basic floating point programs, and all the usual operators are supported, allowing complex trigonometry and mathematical formulae to be implemented and solved.

How does it rate as an educational package? Considering the

number of manuals, the reference card and the wall poster, it seems that Acornsoft knows its market, and everything is produced to support this. Perhaps the only bad point is the fact that two of the valuable paged ROM sockets are taken up by LOGO, and even with the 6502 second processor it is not possible to put one of these on disk. This means that it is likely that both LOGO ROMs will have to be installed inside the machine permanently, no doubt along with Basic and the DFS/NFS, which means the full complement is accounted for. Still, this has always been one of the BBC Micro's problems.



WELCOME TO MY GARDEN
?PRSCREEN

about LOGO: the introductory manual provides an excellent tutorial to LOGO principles and this implementation. The main manual, the reference guide if you like, describes the action of each primitive in a bit more detail and also introduces the more advanced ones. These are the list processing primitives and the machine access primitives, such as the LOGO equivalents of PEEK, POKE and CALL.

Some of the extensions are very useful, such as the screen dump routines for Epson and Olivetti prin-

ters. There are two versions for the Epson; one which uses only black and white and one which represents each colour as a shade. The latter, although a truer representation, is also obviously a lot slower. The drawings in this review were produced using it and an Epson RX-80.

LOGO arithmetic is a little different to standard maths, as floating point numbers are represented in a slightly better way. Whenever we choose to show numbers in exponent format, such as $4.3E2$, it is usual to follow the 'E'

BIZZELL BARGAIN

HIGH SPEED, HIGH QUALITY,
FOUR COLOUR, FLATBED.
PRINTER/PLOTTER Only £399 inc VAT + del

• Prints horizontally or vertically in text.
• Prints four directions and 255 sizes in graphics.
• Plotting area 298x216 mm.
• Plotting speed 100mm/sec.

Ideal for graphs, drawings, symbols, axes, geometric patterns, charts, diagrams, circuits, computer art, flowcharts, 3-D, contours etc. Write with order and cheque (stating computer) to:

BIZZELL COMPUTERS, Walnut Tree House, Fornett St. Peter, Norwich NR16 1HR. (095389-592)

PERSONAL CALLERS WELCOME (10.00-5.00) Phone for appointment

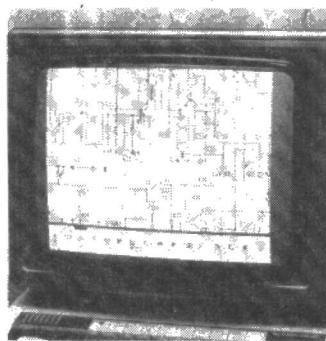
© Centaur Advertising Ltd 01-390 1612

PINEAPPLE SOFTWARE

Programs for the BBC model 'B' with disc drive with FREE updating service on all software

DIAGRAM

- A program which allows you to store very large diagrams - up to 39 mode 0 screens - and view or edit them by SCROLLING the computer screen around over any part of the diagram.



FEATURES

- Draw diagrams, schematics, plans etc., in any aspect ratio, e.g. 10:3, 2:12 screens.
- Access any part of the diagram rapidly by entering an index name, e.g. TR6, R5 etc., to display a specific section of the diagram, and then scroll around to any other part of the diagram using the cursor keys.
- Up to 128 icons may be predefined for each diagram, e.g. Transistors, resistors etc., in full mode 0 definition, up to 32 pixels horizontally by 24 vertically.
- Hard copy printouts in varying print sizes up to 9 mode 0 screens on an A4 size sheet, compatible with most dot matrix printers.
- Many other features including, selectable display colours, comprehensive line drawing facilities, TAB settings, etc.
- Disc contains sample diagrams and two versions of the program, one of which will work from a 16k sideways RAM.
- Comprehensive instruction manual.

£25

Including P & P

Supplied only on disc - 40T/80T compatible

All orders sent by return of post.

**PINEAPPLE SOFTWARE, 39 Brownlea Cdns.
Seven Kings, Ilford, Essex IG3 9NL**

'IMAGE' V.2 - FOR THE ELECTRON/BBC

'Image' was the ultimate tape back-up copier, now it's even better. You can be completely assured that this is the best and most able program of its type available. It can deal with:

- * Locked programs
- * Programs of any length
- * 300 and 1200 BAUD
- * Files
- * ?'s (Ctrl codes) in filename
- * Multiple copies
- * False or trick block info.
- * Changing Filenam
- * Continuous data streams
- * Locking and unlocking programs

It is VERY IMPORTANT INDEED purchasers take note that 'IMAGE' is for sale strictly for making BACK-UPS of your own software for your own use, for protecting your own programs, or as an aid to putting software on disk. Any person found using the program for illegal purposes runs the risk of being prosecuted.

To receive your copy of 'Image', send a cheque or P.O. to the sum of An Astounding £4.80 to:

**PETER DONN, Dept. (EC),
33 Little Gaynes Lane, Upminster, Essex RM14 2JR.**

Please state BBC or Electron version.

V.1 owners can obtain V.2 by sending £1.50 + V.1 without case.

More This Month at Maplin

256K D-RAM 41256 - 150ns ONLY £9.95 (QY74R).

256K EPROM 27256 - 250ns ONLY £18.95 (QY75S).

Right-angle PCB mounting rotary switches: 1P12W - FT56L, 2P6W - FT57M, 3P4W - FT58N, 4P3W - FT59P. All £2.95 each.

Stepper motor 48 steps/rev, 12V 0.13A per phase, 4-phase unipolar, 57g, working torque 8mNm max. ONLY £9.95 (FT73Q).

Driver/chip for motor: SAA1027 ONLY £3.75 (QY76H).

★ **SAVE**★ 1 Kit containing everything you need: motor, SAA1027, data sheet and passives ONLY £13.35 (LK76H).



Sounds Terrific



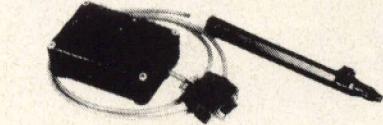
Professional Quality
High Power Loudspeakers
featuring:

- ★ Virtually indestructible high-temperature voice-coil reinforced with glass-fibre.
- ★ 100% heat overload tolerance.
- ★ Advanced technology magnet system.
- ★ Rigid cast alloy chassis.
- ★ Linen or Plastiflex elastomer surrounds.
- ★ 5-year guarantee (in addition to statutory rights).

Prices from £17.97.

Send S.A.E. for our free leaflet XH62S.

Top Ten Kits

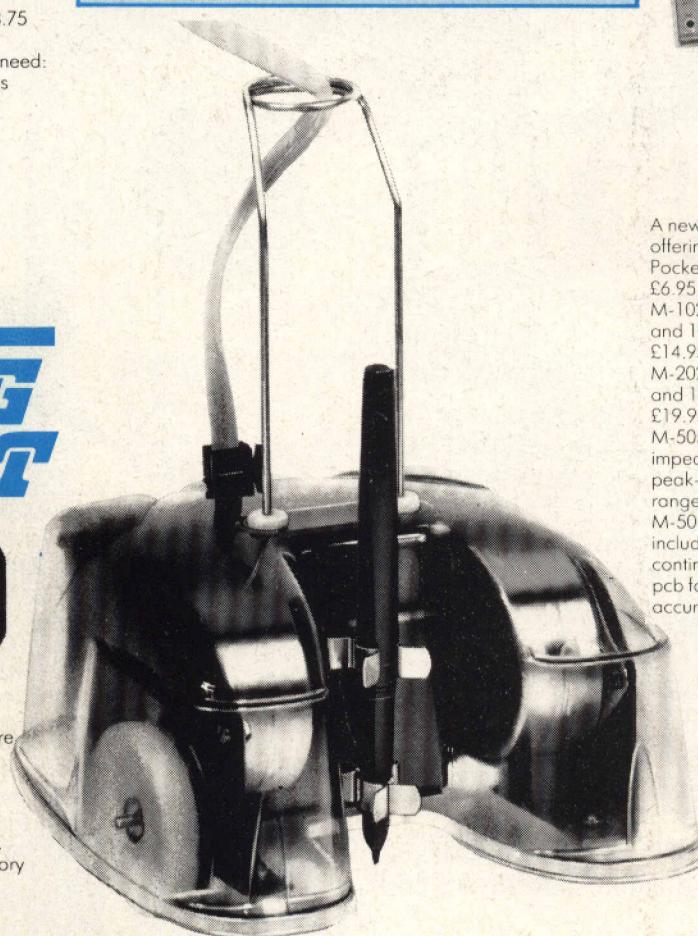


THIS/MONTH	DESCRIPTION	CODE	PRICE	BOOK
1. (-)	Live-Wire Detector	LK63T	£2.95	14 XA14Q
2. (1)	75W Mosfet Amp.	LW51F	£15.95	Best E&MM
3. (2)	Partylite	LW93B	£10.95	Best E&MM
4. (4)	Car Burglar Alarm	LW78K	£7.49	4 XA04E
5. (9)	U/sonic Intruder Dtc	LW83E	£10.95	4 XA04E
6. (10)	Computordrum	LK52G	£9.95	12 XA12N
7. (8)	Light Pen	LK51F	£10.95	12 XA12N
8. (11)	Syntom Drum Synth	LW86T	£12.95	Best E&MM
9. (7)	8W Amplifier	LW36P	£4.95	Catalogue
10. (6)	ZX81 I/O Port	LW76H	£10.49	4 XA04E



Over 100 other kits also available. All kits supplied with instructions. The descriptions above are necessarily short. Please ensure you know exactly what the kit is and what it comprises before ordering, by checking the appropriate Project Book mentioned in the list above.

Is it a turtle? Is it a robot? Is it a buggy? Yes! it's Zero 2.



- May be used by any computer with RS232 facility.
- Stepper Motor controlled.
- Half millimetre/half degree resolution.
- Uses ordinary felt-tip pens.
- Built-in 2-tone horn, line-follower. LED indicators.

The Zero 2 Robot is the first truly micro robotic system available and remarkably it costs less than £80. Complete kit (only mechanical construction required) £79.95 (LK66W). Full details of power supply and simple interfacing for BBC, Commodore 64 and Spectrum, in Maplin Magazine 15 price 75p (XA15R).

MAPLIN
MAPLIN ELECTRONIC SUPPLIES LTD
Mail-order: P.O. Box 3, Rayleigh, Essex SS6 8LR.
Telephone: Southend (0702) 552911
SHOPS

- BIRMINGHAM Lynton Square, Perry Barr, Tel: 021-356-7292.
- LONDON 159-161 King Street, Hammersmith, W6.
Telephone: 01-748 0926.
- MANCHESTER 8 Oxford Road, Tel: 061-236 0281.
- SOUTHAMPTON 46-48 Bevois Valley Road, Tel: 0703-225831
- SOUTHEND 282-284 London Rd, Westcliff-on-Sea, Essex.
Telephone: 0702-554000. Shops closed all day Monday.

More Choice In Multimeters



A new range of very high quality multimeters offering truly amazing quality at the price.

Pocket Multimeter, 16 ranges, 2,000Ω/V DC/AC £6.95 (YJ06G)

M-102BZ with continuity buzzer, battery tester and 10A DC range, 23 ranges, 20,000Ω/V DC £14.95 (YJ07H)

M-2020S with transistor, diode and LED tester and 10A DC range, 27 ranges, 20,000Ω/V DC £19.95 (YJ08J)

M-5050E Electronic Multimeter with very high impedance FET input, 53 ranges, including peak-to-peak AC, centre-zero and 12A AC/DC ranges £34.95 (YJ09K)

M-5010 Digital Multimeter with 31 ranges including 20Ω and 20μA DC/AC FSD ranges, continuity buzzer, diode test, and gold-plated PCB for long-term reliability and consistent high accuracy (0.25% + 1 digit DCV) £42.50 (YJ10L)



The Maplin Service

All in-stock goods despatched same day for all orders received before 2.00 pm.

All our prices include VAT and carriage (first class up to 750g).

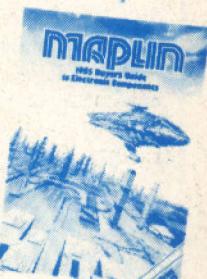
A 50p handling charge must be added if your total order is less than £5.00 on mail-order (except catalogue).



Phone before 2.00 p.m. for same day despatch.

1985 CATALOGUE

Pick up a copy now at any branch of W.H. Smith or in one of our shops. Price £1.35, or by post £1.75 from our Rayleigh address (quote CA02C).



All offers subject to availability.
Prices firm until 10th August 1985.